Semi-supervised clustering applied in revenue accounting

Tianshu Yang

UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE DE DOCTORAT

Le Clustering Semi-supervisé Appliqué à la Comptabilité des Revenus

Tianshu YANG

Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis

Présentée en vue de l'obtention du grade de docteur en Informatique **de** l'Université Côte d'Azur
Dirigée par : Frédéric PRECIOSO
Co-encadrée par : Nicolas PASQUIER
Soutenue le : 14 Décembre 2021

Devant le jury, composé de :
Président du jury :
Pr. Charles BOUVEYRON, Directeur, l'Institut 3IA Côte d'Azur
Rapporteurs :
Mrs. Karell BERTET, Maître de Conférences, La Rochelle Université
Mrs. Anne LAURENT, Professeure, Université Montpellier
Examinateur :
Pr. Bruno CREMILLEUX, Professeur, Université de Caen
Invité :
Mr. Luca MARCHETTI, AMADEUS S.A.S
Mr. Michael DEFOIN-PLATEL, AMADEUS S.A.S

ANRT ASSOCIATION NATIONALE RECHERCHE TECHNOLOGIE

AMADEUS

Abstract

The Amadeus Revenue Accounting workflow automatically process tickets until an error occurs. The workflow is then interrupted and user correction is required on the error. The main problem here is that each error is treated as independent, even if similar errors have already been corrected, resulting in an important waste of efforts. The work of this thesis aims to improve the automation of the error handling process, through clustering of error tickets to form clusters of tickets corresponding to similar anomalies and requiring similar correction processes.

We propose a new semi-supervised consensus clustering approach named Semi-MultiCons to achieve the thesis goal. Semi-MultiCons makes use of supervised information in both the ensemble member generation and consensus process steps, and manages to generate a recommended consensus solution with a relevant inferred number of clusters k based on ensemble members with different k parameter values. The experimental results demonstrate that Semi-MultiCons is able to alleviate the widely reported negative effect related to the integration of constraints into clustering and has remarkable robustness against noisy constraints. Semi-MultiCons is also proved to be able to handle huge industrial datasets and manages to achieve good performance. With the proposed mini batch mode, Semi-MultiCons can give quick, or even real, time response.

A Proof-of-Concept of Semi-MultiCons with Big Data ecosystem and Cloud platform was developed and deployed in real industrial environment. With the PoC, the user is able to explore clusters of similar error tickets, to validate these clusters, as well as to make batch fix or correction per cluster. The action from user will then be used as supervised information to improve the overall quality of Semi-MultiCons clustering result.

Keywords— Clustering, Semi-supervised consensus clustering, Closed sets, Anomalies correction, Revenue accounting

Résumé

Le flux de travail du système de comptabilisation des recettes (Revenue Accounting Workflow) de Amadeus traite automatiquement les tickets comptables jusqu'à ce qu'une erreur se produise. Le flux de travail est alors interrompu et une action de l'utilisateur est requise pour corriger l'erreur. Le principal problème ici est que chaque erreur est traitée comme indépendante, même si des erreurs similaires ont déjà été corrigées, ce qui entraîne une importante perte de temps. Le travail de cette thèse vise à améliorer l'automatisation du processus de traitement des erreurs, par le regroupement des tickets d'erreur pour former des clusters de tickets correspondant à des anomalies similaires et nécessitant des processus de correction similaires.

Nous proposons une nouvelle approche de clustering semi-supervisé par consensus, nommée Semi-MultiCons, pour atteindre cet objectif. Semi-MultiCons utilise des informations supervisées à la fois dans l'étape de génération des membres de l'ensemble de clusterings initiaux et dans le processus de consensus. Cette approche parvient à générer une solution de clustering par consensus recommandée avec un nombre de clusters inféré K pertinent, à partir de clusterings initiaux avec différents nombres de clusters K. Les résultats expérimentaux démontrent que Semi-MultiCons est capable d'atténuer l'"effet négatif", largement rapporté dans la littérature, lié à l'intégration de contraintes dans le clustering et est remarquablement robuste en présence de bruit dans les contraintes. Semi-MultiCons s'avère également capable de traiter de très larges ensembles de données industrielles et parvient à obtenir de bonnes performances. Avec le mode "mini-batch" proposé, Semi-MultiCons peut donner une réponse rapide, voire en temps réel.

Une preuve de concept de Semi-MultiCons avec l'écosystème Big Data et la plate-forme Cloud est développée et déployée dans un environnement industriel opérationnel. Grâce à cette preuve de concept, l'utilisateur est en mesure d'explorer les clusters de tickets d'erreur similaires, de valider ces clusters et d'effectuer des corrections par lot pour chaque cluster. L'action de l'utilisateur sera ensuite utilisée comme information supervisée afin d'améliorer la qualité du résultat du clustering

de Semi-MultiCons.

Mots clés: Clustering, Clustering semi-supervisé par consensus, Ensembles fermés, Correction des anomalies, Comptabilité des recettes

Acknowledgments

First and foremost, I am deeply grateful to my PhD supervisors Nicolas and Frédéric. Thank you for your valuable comments and suggestions during the entire thesis, your reviews on my writings, your assistance at every stage of the research project and your support in difficult times.

I would like to express my sincere gratitude to my colleagues in the Amadeus API team. I really enjoyed the atmosphere and the working environment in the team, and I have learned a lot, as a person, employee and researcher. Special thanks to my tutors Luca, Antoine and Laurent. Thank you for always giving me a hand when I encounter issues.

Also, a special thanks to my friends, Zhongxian, Ying, Mengqian, Quancheng, for your treasured friendship, understanding and love through the years.

Finally, I want to thank my father, for his unwavering support and belief in me.

To my mom in heaven.

Contents

List of Figures

List of Tables

List of Publications

Implementations

[90] Tianshu Yang, *Implementation of the SMC Approach for Semi-supervised Consensus Clustering Based on Closed Patterns*, 2021, https://github.com/lazzyCloud/semi-multicons

International Conferences

[91] Tianshu Yang, Nicolas Pasquier, Antoine Hom, Laurent Dolle, Frédéric Precioso. *Semi-supervised Consensus Clustering Based on Frequent Closed Itemsets* in Proceedings of the CIKM' 2020 29th ACM International Conference on Information and Knowledge Management, pages 3341-3344, Galway, Ireland, October 2020, ACM Association for Computing Machinery, DOI: 10.1145/3340531.3417453 (Acceptance Rate: 18%). Amadeus Intellectual Property Invention Pa tent ID2326WW00 "Clustering Techniques for Revenue Accounting Error-Handling Automation" Defensive Paper.

[92] Tianshu Yang, Nicolas Pasquier, Frédéric Precioso. *Ensemble Clustering Based Semi-Supervised Learning for Revenue Accounting Workflow Management* in Proceedings of the DATA' 2020 International Conference on Data Science, Technology and Applications, pages 283-293, Paris, France, July 2020, SciTePress Science and Technology Publications, DOI: 10.5220/0009883 802830293 (Acceptance Rate: 14%). Amadeus Intellectual Property Invention Patent ID2326WW00 "Clustering Techniques for Revenue Accounting Error-Handling Automation" Defensive Paper.

International Journals

[94] Tianshu Yang, Nicolas Pasquier, Frédéric Precioso. *Semi-supervised Consensus Clustering Based on Closed Patterns* in Knowledge-Based Systems (Impact Factor 8.038), Volume 235, Article 107599, Elsevier, January 2022.

Patents

[95] Tianshu Yang, Nicolas Pasquier, Frédéric Precioso, Antoine Hom, Laurent Dollé. *Clustering Techniques for Revenue Accounting Error-Handling Automation*. Intellectual Property Invention ID2326WW00. Intellectual Property Board, Amadeus S.A.S., Sophia Antipolis, France. Patent number: ID2326WW00, July 2020.

Unpublished

[93] Tianshu Yang, Nicolas Pasquier, Frédéric Precioso. *Semi-Supervised Consensus Clustering Based on Frequent Closed Itemsets* in PhD Track of the IDA'2020 18th International Symposium on Intelligent Data Analysis, Bodenseeforum, Germany, April 2020. PhD Track cancelled due to the COVID-19 pandemic.

Financial Support

This PhD Thesis work was carried out as part of the IDEX UCA^{JEDI} MC^2 joint project between Amadeus S.A.S. and the Université Côte d'Azur. The MC^2 joint project aims at integrating innovative ensemble-based machine learning techniques, including multi-level consensus unsupervised and semi-supervised learning techniques, developed in the I3S Laboratory of Université Côte d'Azur into Amadeus' business optimization processes. This project integrates two central work focusing on this objective with the Amadeus–Université Côte d'Azur joint PhD Thesis of Mrs. Tianshu YANG and the Université Côte d'Azur Post-doctorate of M. Sujoy CHATTERJEE.

The IDEX UCA^{JEDI} MC^2 joint project between Amadeus and the Université Côte d'Azur is supported by the French government, through the UCA^{JEDI} Investments in the Future project managed by the French National Research Agency (ANR) with the reference number ANR-15-IDEX-01.

Chapter 1

Introduction

This chapter explains the related background knowledge and motivation of the work presented in this thesis. It highlights the need to improve the Revenue Accounting Workflow of Amadeus with semi-automatic error handling tools. Further, it explains our objective in two stages. Finally, an overview over the structure of the thesis is given.

1.1 Background and Motivation

Amadeus S.A.S is the leading provider of IT solutions to the global travel and tourism industry. Amadeus creates solutions that enable airlines, airports, hotels, railways, search engines, travel agencies, tour operators and other stakeholders to operate and improve travel management worldwide.

Revenue Accounting (RA) refers to the process of managing and dispatching to the different suppliers involved the amount collected from customer's payment for their travel. This process involves multiple successive treatments of the data in input represented as a ticket calculation code sequence for each travel.

The Amadeus revenue accounting system helps customers performing revenue accounting. It consists of a sequence of modules, referred as Revenue Accounting Workflow (RAW), each one performing a computation from its input and sending its output to the next module. It generates the different amounts related to a journey and the different travels it involves: Interline proration between transportation operators, calculation of fees, commissions and taxes, etc. Computation results of each input ticket calculation code sequence are structured as Business Object Model (BOM) and stored in a database. The Revenue Accounting Workflow is illustrated in Figure 1.1.

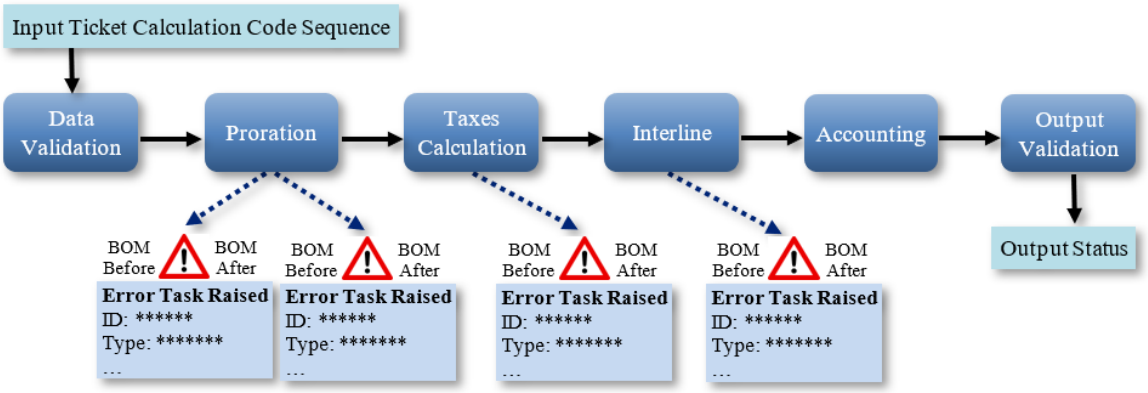


Figure 1.1: Example Revenue Accounting Workflow and error tasks raised by anomalies. Each module in the workflow performs a computation from its input and sends its output to the next module. An error task is raised when the input and/or the output of a module is abnormal.

The first stage of the Revenue Accounting Workflow is to validate input data. Next, amounts are prorated to travel coupon level. Then, taxes, fees, charges and other values are calculated based on these prorated coupon amounts and local government laws. If any travel coupon is operated by other airlines than the seller, an interline process is launched so that involved airlines can negotiate about the fare of coupon. Finally, the accounting module checks if amounts are balanced, which means credit should be equal to debit to avoid calculation errors.

This process entails complex management constraints and is automated unless an error occurs. Errors, defined by domain experts, refer to situations where the input and/or the output of a module is abnormal. Such anomalies are identified by checking the BOM values before and after each module execution to generate error resolution tasks, described by their associated error ticket. During each module computation, one or several anomalies, such as an incorrect amount computed due to erroneous values in input for example, can occur.

The main problem with the current Tasks Handling Module (THM) is that each task is treated as independent, even if similar errors have already been corrected. The analysis of a sample of 2 000 tasks has shown up to 40% similar tasks. This results in an important waste of efforts and machine learning techniques are considered to help in decreasing costs and time spent on similar error tickets due to their required individual correction.

1.2 Objective

The application of machine learning techniques aims to improve the automation of the task handling process with the automatic identification of anomaly patterns in the Revenue Accounting Workflow, and the automatic or semi-automatic, depending on the type of the anomaly pattern, correction of the task. This application involves the two main steps described hereafter.

The first step is the identification of relevant anomaly patterns, regarding anomaly distinctive features, through the clustering of error tickets to identify clusters of tickets corresponding to similar anomalies and requiring similar correction processes. Error tickets containing information about the transportation coupons of a travel are grouped into clusters corresponding each to a type of anomaly such as a interline calculation or a proration anomaly as illustrated in Figure 1.2.

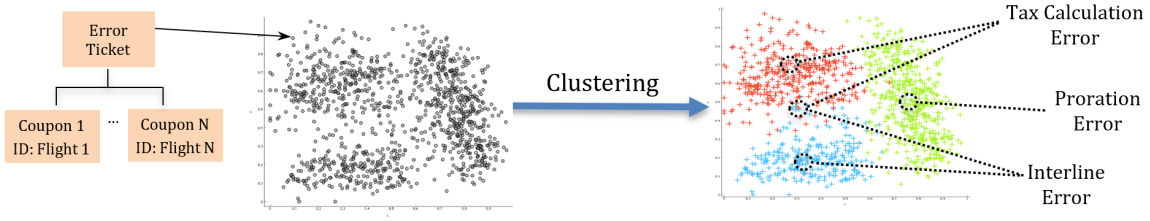


Figure 1.2: Clustering anomaly pattern correction tasks. The clustering of error tickets aims to identify clusters of tickets with analogous anomalies that require similar correction tasks.

The second step is the learning of the correction processes associated to each cluster of tickets, by the assignment of correction actions taken by the users, for the automation of the error correction process. By this assignment, anomaly corrections can be defined for each type of error pattern corresponding to a cluster of error tickets. As illustrated in Figure 1.3, these correction processes can require the intervention of the end-user.

1.3 Outline

This thesis report is organized as follows:

Chapter 2 reviews the central issues of applying classical clustering approaches on Amadeus Revenue Accounting Workflow data, and the most recent algorithmic developments to address these issues. Further, the concepts and challenges in the research field of these developments, which are adapted to Amadeus Revenue Accounting Workflow in the context of this thesis, are presented.

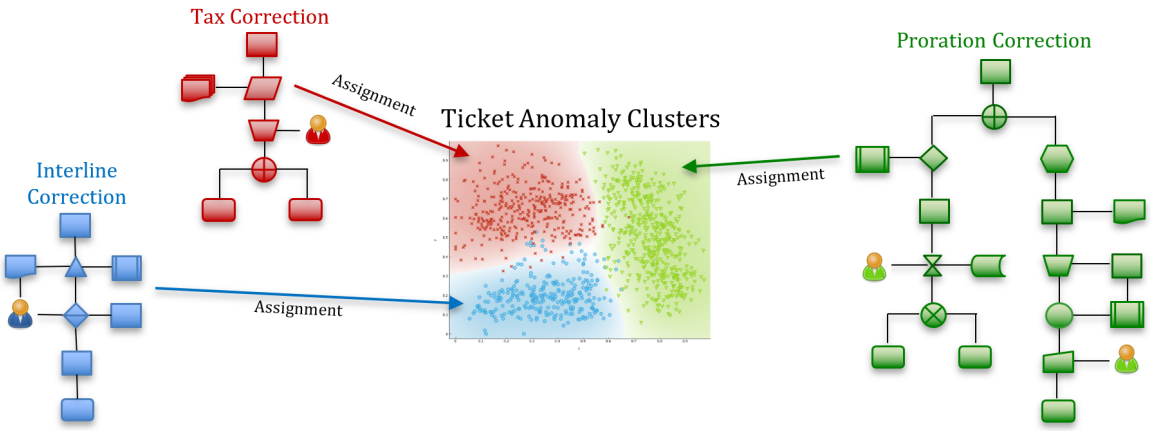


Figure 1.3: Assignment of Correction Processes for Anomaly Pattern Clusters. Each anomaly pattern cluster identified in the previous step is assigned a correction process, which can require the intervention of the end-user.

Chapter 3 shares the central preliminary knowledge about multi-level consensus clustering and the use of pairwise constraints as supervised information in semi-supervised clustering to help in understanding the proposed framework.

Chapter 4 presents the proposed Semi-MultiCons semi-supervised multiple consensus clustering framework. Semi-MultiCons aims to improve the multi-level consensus clustering result by integrating supervised information in the consensus creation process and infer the number of clusters k using frequent closed itemsets extracted from the initial clustering ensemble members.

Chapter 5 introduces the datasets and experimental settings that were used for the eventual experiments presented in the thesis report. These datasets fall into two categories. The first includes benchmark UCI clustering datasets allowing to compare the Semi-MultiCons approach with other state-of-the-art approaches in the literature. The second is a dataset that was created based on Amadeus operational data to evaluate applicability, performance and relevance of the Semi-MultiCons method in the context of realistic use cases.

Chapter 6 analyses the result of the Semi-MultiCons approach on the study datasets, in comparison with other semi-supervised and/or consensus clustering algorithms. Central results show that, without requiring the correct number of clusters k in input, Semi-MultiCons is able to infer this number k and to make a relevant use of supervised information in the form of must-link/cannot-link

constraints. We also show how the final result of Semi-MultiCons, represented as a hierarchical structure of clusters, helps the user to better understand the data space properties and clustering process. Experimental results show that Semi-MultiCons can detect the optimal k in most cases with an equivalent performance as other semi-supervised and/or consensus clustering approaches with this k value provided in input.

Chapter 7 demonstrates the successful applications of the proposed Semi-MultiCons approach. A Proof-of-Concept prototype shows the practical potential of Semi-MultiCons as an error handling module for Amadeus Revenue Accounting Workflow. The prototype automatically groups tasks into clusters requiring similar correction processes and proposes clusters to users. Furthermore, the user can validate these clusters and make batch fix or correction on cluster tickets. The feedback from user will then be used as supervised information to improve the overall quality of further clusterings.

Chapter 8 is an extended analysis about different ways of generating constraints from supervision information and their impact on the robustness of current state-of-the-art semi-supervised clustering models as well as on the Semi-MultiCons approach. Different scenarios for which each approach is more suitable are highlighted.

Chapter 9 summarizes our contributions and concludes the report with remaining open questions. Furthermore, potential improvements and perspectives for future work are proposed.

Chapter 2

Central Issues and Related Work

Clustering, or unsupervised classification, is the computational process that aims to discover clusters (groups) of instances in a dataset. A cluster is a set of instances (e.g., individuals) that are as much as possible similar among themselves within the group and different from one group to another regarding their features represented as variable values. See [27], [46] and [85] for surveys of clustering algorithms. Directly applying classical clustering approaches on Revenue Accounting Workflow data shows several central issues:

Algorithmic Configuration Choice Issue: Different algorithmic configurations, i.e., a specific algorithm with a specific parameterization, can provide different clustering solutions. Hence, each algorithm relies on a particular assumption regarding the distribution model of instances in the data space, and each parameterization defines a manner to put in practice this model. The quality of the resulting clustering will depend to which extent they are adequate to the analyzed data space properties, as studied in [38] and [86].

Clusters Internal Validation Issue: A distinctive characteristic of clustering applications, regarding classification issues, i.e., to distinguish application classes, is the absence of initial prior knowledge on the data space properties and of labelled (class annotated) data to help choosing an algorithmic configuration that is appropriate for the analysed dataset. Moreover, the problem of choosing an adequate algorithmic configuration and obtaining a meaningful clustering is exacerbated by the existing difficulty of objectively analysing the quality of the clusters obtained. If several internal validation measures have been proposed, each measure also relies on a specific assumption of the distribution model of instances in the data space and can thus overrate clustering

results of algorithms based on the same model (e.g., centroid or density based). See [15], [37], [60] and [68] for studies on clustering validation measures.

Clusters to Application Classes Issue: The objective of the assignment of application classes (e.g., anomaly correction classes) to clusters is to connect the clusters and the correction classes in order that each cluster is as much as possible representative, i.e. distinctive in the data space, of an application class. Classical unsupervised clustering techniques are not able to make use of any supervised information and thus cannot address this issue.

These central issues, especially the clusters to application classes issue, imply the development of semi-supervised algorithmic solutions combining unsupervised internal validation of consensus clusters and supervised external validation of consensus clusters based on Amadeus business metrics.

2.1 Semi-supervised Clustering

Semi-supervised clustering incorporates prior knowledge such as class labels or pairwise constraints into classical clustering methods to obtain better quality result. In the context of this thesis, supervised information is used to help solving clusters to application classes issue, as well as to enforce clustering approach for producing clusters meeting the application constraints and/or user preferences. The recent semi-supervised clustering algorithms can be classified into three types [6]: constraint-based methods, distance-based methods and hybrid methods.

- **Constraint-based methods** refers to algorithms that utilize supervised information to restrict the feasible solutions when assigning instances to clusters, either directly by changing assignment strategy to prevent assignment that violates supervised information [75], or indirectly by penalizing and/or rewarding objective function if supervised information is violated and/or satisfied [17].

- **Distance-based methods** means supervised information is applied in distance learning. This distance can be a distance in the original data space [83], a distance in low dimension feature space [67] or even a kernel distance matrix [39].

- **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 application constraints and user preferences. However, since

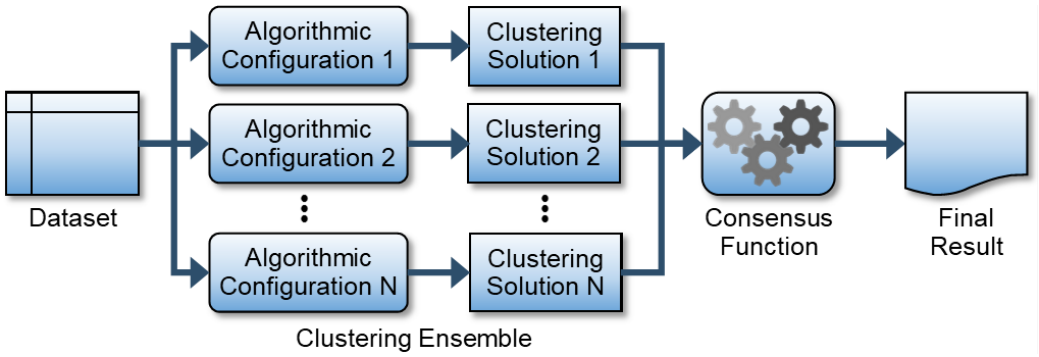


Figure 2.1: Overview of consensus clustering procedure. Different clustering algorithmic configurations are applied to the input dataset for creating the clustering ensemble to which is applied the consensus function for generating the final clustering result.

most semi-supervised algorithms are iterative and sensitive to input order of data, they can encounter problems of stability and robustness.

2.2 Ensemble Clustering

We present hereafter the ensemble clustering technique that tackles the problem of choosing algorithmic configurations. Ensemble clustering, or consensus clustering, approaches aim to address the limitations of single clustering approaches and to improve the robustness and quality of clustering result by combining multiple clustering solutions. These initial solutions, called base clusterings, are each generated from a different clustering algorithmic configuration, i.e., different algorithms with different parameterizations are used, for generating more robust consensus clusters corresponding to agreements between base clusters [69].

The problem can be described as follows. Let m denote the number of clustering solutions. Φ represents the result sets of clustering solutions $\Phi = \{\gamma^1, \gamma^2, ..., \gamma^m\}$, where $\gamma^i = \{C_1^i, C_2^i, ..., C_k^i\}$ 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, ensemble clustering methods consist of two stages. At the first stage, different clustering solutions are generated from the same dataset. Then, in the second stage, a consensus function is applied on these clustering solutions to find the final consensus partition [10]. Figure 2.1 shows an overview of the ensemble clustering process.

Existing ensemble clustering approaches can be classified into the four following categories:

- Approaches considering the clustering ensemble problem as a clustering of categorical data.
- Approaches based on the generation of an instance co-association matrix depicting the number of assignments of each pair of objects to the same cluster in a clustering solution.
- Approaches that rely on the generation of a cluster association matrix based on the number of objects that were commonly assigned to the clusters in a clustering solution.
- Approaches that consider the problem as a graph, or hypergraph, partitioning problem.

Ensemble clustering is proved to outperform single clustering methods on stability and accuracy [80, 70]. However, it has some limitations in the context of this thesis work. Indeed, most ensemble clustering approaches require the user to define the number of clusters to generate prior to the execution. Moreover, approaches based on instance to instance relationship analysis require to generate large association matrices (N^2 size for N instances) which is unfeasible for very large datasets (e.g., millions of objects) due to space and time complexities of the matrix computation and manipulation. Also, as an unsupervised learning method, it is not designed to use any supervised information, even though sometimes a small part of such information is available. See [9], [44] and [73] for extensive reviews and studies on ensemble clustering algorithmic approaches.

2.3 Semi-supervised Ensemble Clustering

Considering the limitations of semi-supervised clustering and ensemble clustering, it is natural to combine them, and thus semi-supervised ensemble clustering emerged. Semi-supervised ensemble clustering not only consider supervised information, but also integrate multiple clustering results into a unified consensus solution to improve the quality, stability and robustness of the final result [97].

Supervision information can be used in both steps of consensus clustering as shown in Figure 2.2. 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 with no difference, the existence of supervision information makes it possible to assign different weights to clusters. Usually, clusters that violate supervision information are eliminated or assigned a lower weight.

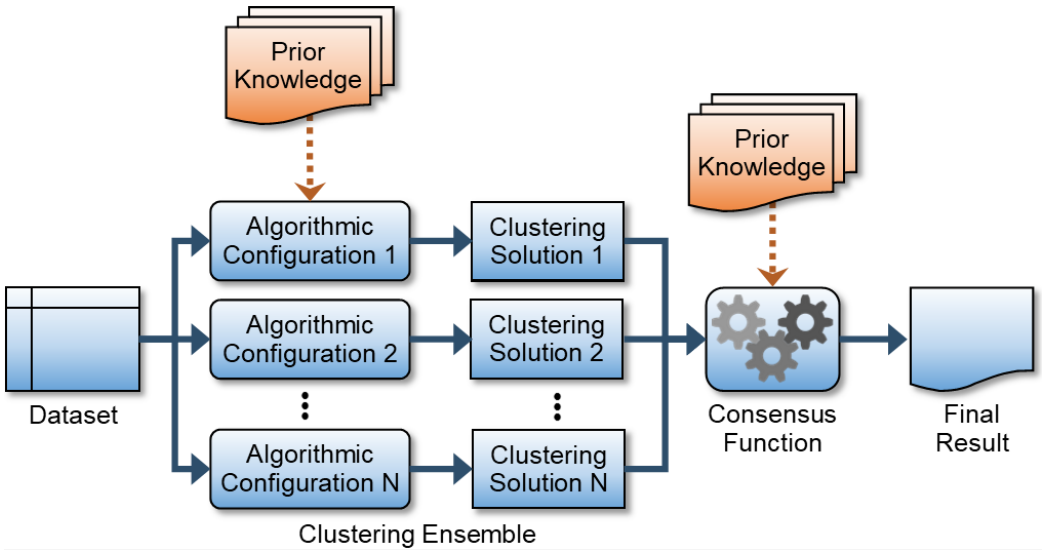


Figure 2.2: Overview of semi-supervised consensus clustering procedure. Supervised information is used as prior knowledge either or both during the clustering ensemble creation and the generation of the final clustering result by the consensus function.

Several semi-supervised consensus clustering approaches have been proposed during the last few years. In [99], 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 a spectral clustering based consensus function. In [42], voting based consensus clustering is extended to semi-supervised by replacing unsupervised base clusterings with semi-supervised base clusterings. The user is 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 [96], an improved COP-Kmeans (Constraint Partitioning K-means) algorithm [75] is proposed as base semi-supervised clustering method and a new constrained self-organizing map as consensus function. In [80], a semi-supervised spectral clustering is applied as base clustering, while their graph based consensus function, called Hybrid Bipartite Graph Function [29], remains unsupervised. In [51], a genetic algorithm based consensus function is extended by taking supervision information into consideration in the fitness function. In [98], cluster ensemble members are generated by a constraint propagation algorithm [50], which is a semi-supervised method. The prior knowledge is also used to evaluate and eliminate ensemble members. Only a subset of these ensemble members is taken into account in the graph based consensus function. In [82], prior knowledge is integrated to improve the CHAMELEON unsupervised hierarchical clustering algorithm [45] and

Table 2.1: A summary on semi-supervised consensus clustering methods. For each referred method, the approach used for base clustering generation and consensus function, and the steps involving the use of supervised information are described.

Method	Base clustering	Consensus function
[99]	Spectral clustering	Spectral clustering
[42]	Semi-supervised clustering	Voting based approach
[96]	Improved COP-Means	COP-SOM
[80]	Semi-supervised spectral clustering	Hybrid Bipartite Graph Function
[51]	Unsupervised clustering	Genetic algorithm
[98]	Constraint propagation	Normalized cut
[82]	Improved CHAMELEON	Co-association based approach
[97]	Constraint propagation	Weighted normalized cut

in the co-association matrix used by the consensus function. In [97], instead of using all prior knowledge, a different subset of prior knowledge is assigned 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 these related work is shown in Table 2.1.

2.4 Multi-level Clustering

In the context of the Revenue Accounting Workflow management, one class of correction processes can correspond to several error ticket clusters, and each cluster can correspond to several correction process classes. In the thesis, the use of multi-level clustering techniques aims to discriminate the application classes according to their properties in the data space, and potentially refine them by distinguishing different sub-classes of a class according to the different modeling properties of each cluster in the data space.

Multi-level clustering generates a hierarchical decomposition of clusters, where a cluster at a level in the hierarchy can be decomposed into several smaller clusters in the sub-levels of the hierarchy. Such a clustering approach can provide a relevant framework for the identification of correction process classes and sub-classes as illustrated in Figure 1.3, where the proration correction process is divided into two sub-classes corresponding to two sub-clusters in the data space [28].

2.5 Proposed Solution to Central Issues

The proposed framework is a combination of semi-supervised clustering, ensemble clustering and multi-level clustering. With semi-supervised clustering, we can use available supervised information to improve clustering quality, to assign application classes to clusters as well as to validate model performance. Ensemble clustering technique simplifies the algorithmic configuration choice. Results from multiple clustering approaches with different algorithmic configurations are combined to generate more robust consensus clusters. The use of multi-level clustering technique makes it possible to refine clusters according to user preferences and application objectives. See Chapter 4 for more details of the proposed framework.

Chapter 3

Preliminaries

The proposed approach relies on the use of partial initial 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 chapter.

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 [78], 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 [74]:

- A must-link constraint implies that two instances must be assigned to the same cluster, or more generally, they are more likely to be similar with each other.
- A cannot-link constraint implies that two instances cannot be assigned to the same cluster, or more generally, they are more likely to be dissimilar with each other.

Must-link and cannot-link constraints have several variations. [17] proposes δ-*Constraint* and ϵ-*Constraint*. [57] mentions the notion *Interval Constraints* and *Non-Interval Constraints* to respec-

tively find clusters which are continuous intervals and/or which are not intervals on a dimension in co-clustering task applied on genes.

Must-link constraints and cannot-link constraints are widely used in semi-supervised clustering algorithms [74, 61], in metric learning [79], in dimensionality reduction [11] and a lot of other domains. In this work, prior knowledge is provided in the form of must-link and cannot-link constraints. Classical supervised information such as class labels can be translated into pairwise constraints.

3.1.1 Class Labels vs. Pairwise Constraints

Constraints are easier to obtain than class labels while constraints provide less information than labels [12]. Assume that we have three instances x, y and z, where x and y have class label A while z has class label B. Three constraints can thereby be conjectured: Must-link constraint between x and y, cannot-link constraint between x and z, and cannot-link constraint between y and z. By defining number of clusters $k = 2$, only one possible partition satisfies all constraints: $\Pi_1 = \{x, y\}$, $\Pi_2 = \{z\}$. From this example, we can see the possibilities to transform class labels into constraints. Note that, the user does not need to know explicitly what are the labels of x, y and z if he is only required to provide constraints. However, the final label of partitions is ambiguous without explicit information about A and B.

3.1.2 Must-Link Constraint Properties

Must-link constraint is symmetrical, reflexive and transitive.

- Symmetrical: x has a must-link constraint with itself x.
- Reflexive: If x has a must-link constraint with y, then y has a must-link constraint with x.
- Transitive: If x has a must-link constraint with y and y has a must-link constraint with z, then x has a must-link constraint with z.

Also, a set of instances such that each is connected to the other via an explicit or implied must-link constraint is called Connected Component [16]. For example, if x has two must-link constraints, with y and with z, then $\{x, y, z\}$ is a connected component.

3.1.3 Transitive Inference of Must-Link Constraints

The transitive property of must-link constraints and the notion of connected components permit us to infer more must-link constraints based on the original constraint set [8, 75, 19].

- Let CC_i be a connected component among original constraints. Then for every pair of instances (x, y) in CC_i, a must-link constraint between x and y can be inferred if it does not appear in the initial constraint set.
- Let CC_i and CC_j be two connected components constructed from the original constraints. If there exists a must-link constraint between x and y, where $x \in CC_i$ and $y \in CC_j$, then for all $a \in CC_i$ and $b \in CC_j$, there must exist a must-link constraint between a and b, which means CC_i and CC_j can be merged into one connected component.

3.1.4 Cannot-Link Constraint Properties

Unlike must-link constraints, cannot-link constraints only have reflexive property, since if cannot-link constraints between x and y and between y and z are given, then it is not guarantee that x and z cannot-link to each other.

- Reflexive: If x has a cannot-link constraint with y, then y has a cannot-link constraint with x.

3.1.5 Transitive Inference of Cannot-Link Constraints

Even though cannot-link constraint type does not have as many properties as must-link constraint type, additional cannot-link constraints can still be inferred by combining both [8, 75, 19].

- Let CC_i and CC_j be two connected components constructed from original constraints. If there exists a cannot-link constraint between x and y, where $x \in CC_i$ and $y \in CC_j$, then for all $a \in CC_i$ and $b \in CC_j$, there must exist a cannot-link constraint between a and b.

3.1.6 Conclusion

Applying constraint properties to compute a full set of constraints is necessary and compulsory before employing any clustering method. This step guarantees that no indirect/inferred constraint is missing, or will be violated unintentionally during the training process.

Table 3.1: Example of clustering ensemble members. Each of the five clustering results γ_i, for i between 1 and 5, is represented as the list of clusters C_i^j with their respective list of instances.

Base clustering	List of base clusters with instance set		
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 MultiCons Approach

The proposed approach extends the MultiCons multiple consensus clustering approach proposed in [1]. Based on frequent closed itemset mining technique, MultiCons is able to discover frequent closed patterns among different base clustering solutions. Each frequent closed pattern defines the agreement of a subset of clusters in partitioning a set of instances. By dividing/merging these patterns into groups, MultiCons generates multiple consensuses in a tree-like structure that helps understanding the clustering process and the data space subjacent inherent structures.

In the following sections, the MultiCons approach is explained step by step by using a dataset $X = \{x_1, x_2, ..., x_9\}$ as a support example.

3.2.1 Ensemble Members

The first step of MultiCons is to generate ensemble members. Different clustering algorithms and different parameters can be used in this step as user preference. Consider five unsupervised clustering methods are applied on dataset X to generate the five ensemble members shown in Table 3.1.

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.

Table 3.2: Example binary membership matrix *M*. Each clusters C_i^j of clustering result γ_i in Table 3.1 is represented as a 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

The binary membership matrix *M* for the support example in Table 3.1 is shown in Table 3.2. 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 are extracted from the binary membership matrix using the *apriori()* function of the R package *arules* [36] with the "closed frequent itemsets" parameter. A closed pattern is 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 *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, that is 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 patterns 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 $N.M$, where M is the number of base clusters, with $M \ll N$, and regarding the demonstrated scalability

Table 3.3: Closed patterns extracted from example binary membership matrix *M*. The seven closed patterns extracted from the binary membership matrix in Table 3.2 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	$\{C_3^1\}$	$\{x_1, x_2, x_3, x_4, x_5\}$

properties of Galois closed set extraction algorithms [7, 52, 88].

For further processing, 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. The instance set can also be regarded as a cluster. The extracted closed patterns for the support example are shown in Table 3.3.

3.2.4 Consensus Function

In [1], five consensus functions can be applied on instance sets of Table 3.4 to obtain the final consensus. Here, we are interested in an iterative merging/splitting approach proposed in [2].

The consensus function in [2] generates L_{unique} consensus solutions, where L_{unique} is the number of unique values among the cluster set sizes $|L_i|$. Concretely, we iterate the loop index l_t from the maximum value of $|L_i|$ to the minimum value of $|L_i|$ and generate one consensus solution S_{l_t} per iteration. The consensus solution S_{l_t} is generated based on instance sets with $|L_i| = l_t$ and the consensus solution $S_{l_{t-1}}$ of the previous iteration.

For each consensus solution, the clusters must be disjoint. Therefore, the consensus function determine to either merge or split two intersecting cluster sets X and Y using Jaccard similarity [43]:

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

If $s(X, Y)$ is greater than the input merging threshold MT, usually set to 0.5 by default, then they are merged; otherwise the larger cluster 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 an important majority of the benchmark datasets, originating from different application contexts and with different data space structure properties, that have been tested. The consensus process repeats until there is no intersection among clusters. The pseudo-code of this 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 clustering solutions S_l

$L_{unique} \leftarrow$ unique values in the list of pattern lengths $|L_i|$
$S_{l_0} \leftarrow \emptyset$
for l_t in L_{unique} **do**
    $S_{l_t} \leftarrow S_{l_{t-1}} \cup C_i$ which length $|L_i| = l$
    endFlag $\leftarrow$ *True*
    repeat
        for each pair of clusters (X, Y) in S_{l_t} **do**
            if X intersects with Y **then**
                endFlag $\leftarrow$ *False*
                Calculate $s(X, Y)$
                if $s(X, Y) >= MT$ **then**
                    Merge X and Y
                else
                    Split larger cluster
                end if
            end if
        end for
    until *endFlag* = *True*
end for

The execution of the closed pattern based consensus function for the support example is shown in Table 3.4. The iterations consider closed patterns in decreasing order of their cluster set size $|L_i|$. Each 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

Table 3.4: Example MultiCons consensus process from closed patterns in Table 3.3. Closed patterns are processed in decreasing order of their cluster set size $l_t = |L_i|$, and the first line of each row shows the patterns considered during the l_t 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_t} of iteration l_t is shown on the last line of the row.

Size l_t	Set $S_{l_{t-1}} \cup C_i$ patterns with $\|L_i\| = l_t$	Processing explanation
5	$S_{l_0} = \emptyset$, 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 $\|L_i\| = 4$: $\{\{x_4, x_5, x_6, x_7\}\}$	
	$\{\{x_1, x_2, x_3\}, \underline{\{x_4, x_5\}}, \{x_6, x_7\}, \{x_8, x_9\}, \underline{\{x_4, x_5, x_6, x_7\}}\}$	s(X, Y) = 1, merge
	$\{\{x_1, x_2, x_3\}, \underline{\{x_4, x_5, x_6, x_7\}}, \underline{\{x_6, x_7\}}, \{x_8, x_9\}\}$	s(X, Y) = 1, merge
	$\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$	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 $\|L_i\| = 2$: $\{\{x_4, x_5, x_6, x_7, x_8, x_9\}\}$	
	$\{\{x_1, x_2, x_3\}, \underline{\{x_4, x_5, x_6, x_7\}}, \{x_8, x_9\}, \underline{\{x_4, x_5, x_6, x_7, x_8, x_9\}}\}$	s(X, Y) = 1, merge
	$\{\{x_1, x_2, x_3\}, \underline{\{x_4, x_5, x_6, x_7, x_8, x_9\}}, \underline{\{x_8, x_9\}}\}$	s(X, Y) = 1, merge
	$\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}\}$	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 $\|L_i\| = 1$: $\{\{x_1, x_2, x_3, x_4, x_5\}\}$	
	$\{\underline{\{x_1, x_2, x_3\}}, \{x_4, x_5, x_6, x_7, x_8, x_9\}, \underline{\{x_1, x_2, x_3, x_4, x_5\}}\}$	s(X, Y) = 1, merge
	$\{\underline{\{x_1, x_2, x_3, x_4, x_5\}}, \underline{\{x_4, x_5, x_6, x_7, x_8, x_9\}}\}$	s(X, Y) = 0.4, split the larger cluster
	$\{\{x_1, x_2, x_3, x_4, x_5\}, \{x_6, x_7, x_8, x_9\}\}$	No intersection, generate S_1
End	$S_1 = \{\{x_1, x_2, x_3, x_4, x_5\}, \{x_6, x_7, x_8, x_9\}\}$	

and Y considered, i.e., among the previously generated clusters and the considered closed patterns, that are intersecting are underlined in the execution trace. The last row shows the S_1 consensus clustering at the top of the hierarchical graphical representation in Figure 3.1. For simplification of the presentation, the iterations of the loop with non-overlapping X and Y are omitted.

3.2.5 Hierarchical Graphical Representation

The generated consensus solutions are presented to the user in a hierarchical graphical representation, as shown in Figure 3.1 for the support example. Each level represents a generated consensus solution corresponding to a given minimal number of agreements between the base clusterings. This minimal number of agreements corresponds to the cluster set size of closed patterns considered during the iteration that generated the solution. A consensus solution that is generated by several

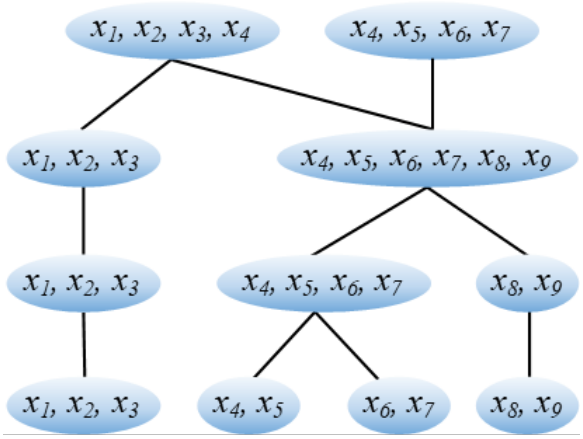


Figure 3.1: Hierarchical consensus clustering of the MultiCons approach. Each level in the hierarchical representation depicts a clustering of the dataset, where nodes represent instance sets of clusters and edges represent inclusion relationships between clusters. This representation shows the successive groupings of instances, corresponding each to a different number of agreements between clusterings of the ensemble.

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 the figure), 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 clustering result depending on objectives, requirements and constraints of the application performed.

3.2.6 Properties of the MultiCons Approach

MultiCons has several advantages over other consensus clustering methods, including:

- No limitation on the selection of the base clustering algorithms and/or their settings. Any algorithm and any setting can be used.
- The number of cluster K is not required as a parameter for the final consensus solution since MultiCons is able to discover automatically the internal structure of the hidden clusters.
- The search for the consensus solution is performed on a pattern-based space instead of the data instances space, thus highly pruning the search space.

- The process of building the consensus solutions is presented in a hierarchical view, which provides significant demonstration of the relationships among instances in the data space.
- MultiCons generates multiple candidate clustering consensuses instead of a single one, allowing the user to select the most appropriate solution regarding possible application constraints and requirements.

However, due to the fact that the consensus process needs to be repeated until all clusters are disjoint, it is hard to evaluate the complexity of MultiCons, which strongly relies on how many times the process is repeated.

Chapter 4

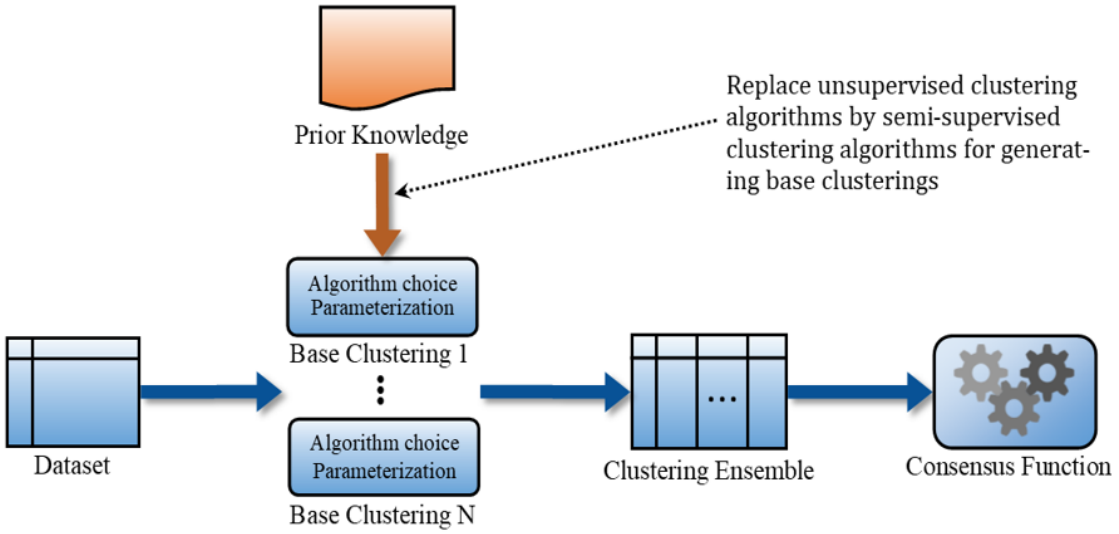
Semi-Supervised Multiple Consensus Clustering

Studies of the Revenue Accounting Workflow problem, the semi-supervised learning concepts and the MultiCons approach lead to the development of a new closed pattern-based consensus semi-supervised algorithmic approach named *Semi-MultiCons*. This approach integrates constraints in different phases of the consensus clustering process.

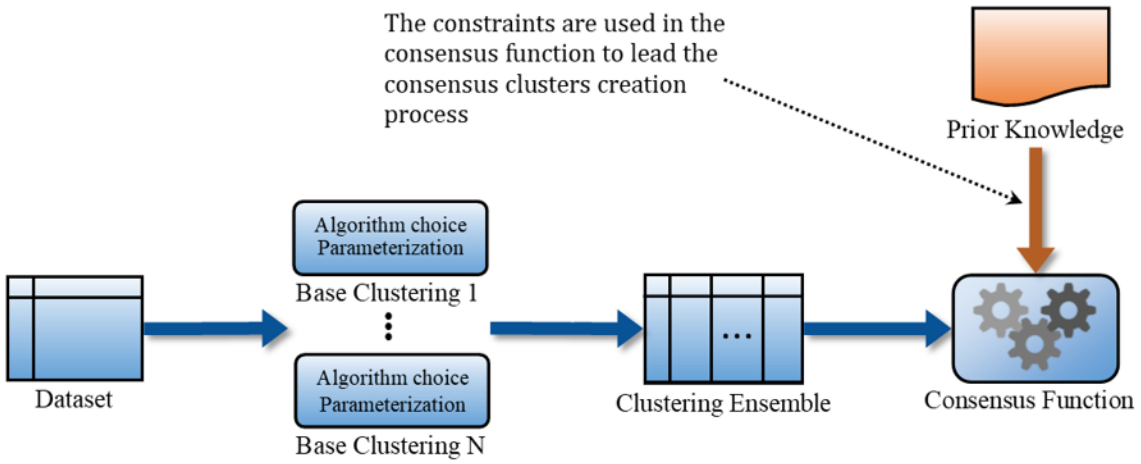
As depicted in Figure 4.1, cannot-link and must-link constraints can be integrated during the creation of the base clusterings, by using semi-supervised clustering algorithms. Cannot-link and must-link constraints can also be integrated during the processing of the clustering ensemble by the consensus function to generate consensus clusters, so that the resulting consensus clusterings comply as far as possible with the integrated constraints.

In the Semi-MultiCons approach, the implementation of the consensus function is optimized, compared to the MultiCons approach, by reducing 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 were also developed. These new algorithmic processes introduce the use of supervised information, represented as must-link and cannot-link constraints, to optimize the relevance of the recommended consensus solution regarding available prior knowledge.

The workflow of the Semi-MultiCons approach is shown in Figure 4.2. 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.



(a) Constraint integration during the creation of the base clusterings.



(b) Constraint integration during the processing of the clustering ensemble by the consensus function to generate consensus clusters.

Figure 4.1: Framework of the Semi-MultiCons approach. Constraints can be integrated during the creation of the base clusterings and/or during the processing of the clustering ensemble by the consensus function to generate consensus clusters.

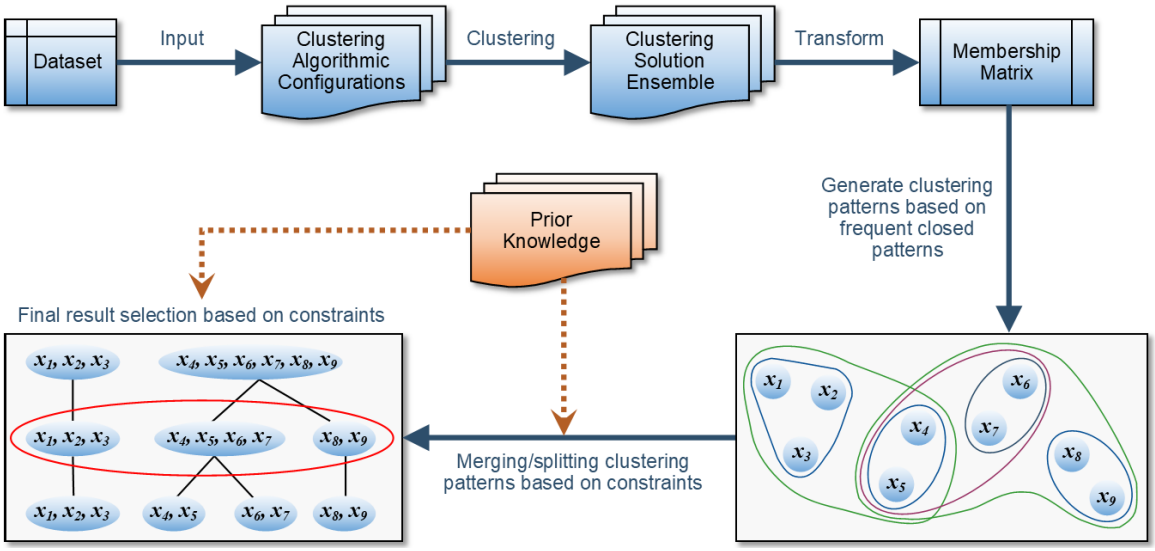


Figure 4.2: Workflow of the Semi-MultiCons approach. The dataset in input is processed by different clustering algorithmic configurations to create the clustering ensemble. This ensemble is transformed to a binary membership matrix from which closed patterns are extracted. These closed patterns are then processed, according to supervised information, by the Semi-MultiCons consensus function to generate the hierarchical consensus clustering result in which the final recommended clustering is identified.

These steps define the search space for the consensus function and the generation of the hierarchical consensus clustering graphical representation. Semi-MultiCons consists of the following four central phases:

1. Ensemble member generation.
2. Closed pattern extraction from the binary membership matrix.
3. Constraint-based consensus function.
4. Recommended solution selection

These phases, as well as the related optimizations compared to the MultiCons approach and the pseudo-code of the consensus function of the Semi-MultiCons approach, are presented in the following sections.

4.1 Ensemble Member Generation

MultiCons uses unsupervised clustering algorithms to generate base clustering solutions, while in Semi-MultiCons, the choice is extended to semi-supervised clustering. Both unsupervised clustering and semi-supervised clustering can be used in ensemble member generation step of Semi-MultiCons.

4.2 Binary Membership Matrix and Closed Pattern Extraction

Semi-MultiCons applies the same closed patterns extraction technique as MultiCons. See Section 3.2.2 and Section 3.2.3 for a concrete example.

4.3 Implementation Optimization

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

The implementation is optimized by handling separately the closed patterns C_i with $|L_i| = l_t$ and already generated clusters in S_{l_t}, 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 $|L_i| = l_t$, intersecting cluster sets Y in S_{l_t} are enumerated, and merge/split operation is performed 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_t}. Cluster X is then added to S_{l_t} and the function continues to enumerate closed patterns C_i with $|L_i| = l_t$. 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_t} 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 approximate complexity of $O(kN^2)$ on average for the Semi-MultiCons consensus process.

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_l

$L_{unique} \leftarrow$ unique values in the list of pattern lengths $|L_i|$
$S_{l_0} \leftarrow \emptyset$
for l_t in L_{unique} **do**
    $S_{l_t} \leftarrow S_{l_{t-1}}$
    $C_{now} \leftarrow C_i$ with length $L_i = l_t$
    for X in C_{now} **do**
        for Y in S_{l_t} **do**
            if Should merge according to constraint-based consensus function **then**
                $X \leftarrow X \cup Y$
                $Y \leftarrow \emptyset$
            else
                Larger cluster $\leftarrow$ larger cluster - $X \cap Y$
            end if
        end for
        Add X to S_{l_t}
    end for
end for

4.4 Constraint-based Consensus Function

After closed patterns are constructed, Semi-MultiCons combines together these patterns using an iterative merging/splitting approach based on constraints. The novel Semi-MultiCons constraints-based consensus function makes use of supervised information represented in the form of must-link 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 merging two clusters X and Y is defined as:

$$S_{merge} = \frac{\sum_{a \in X \setminus (X \cap Y)} \sum_{b \in Y \setminus (X \cap Y)} g(a,b)}{|(X \cup Y) \setminus (X \cap Y)|} \tag{4.1}$$

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

$$S_{splitX} = \frac{-\sum_{a \in X \setminus (X \cap Y)} \sum_{b \in X \cap Y} g(a,b)}{|X|} \tag{4.2}$$

$$S_{splitY} = \frac{-\sum_{a \in Y \setminus (X \cap Y)} \sum_{b \in X \cap Y} g(a,b)}{|Y|} \tag{4.3}$$

If the three scores all equal to 0, it means that no supervised 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 supervised information in the consensus generation process, let's consider the support example consisting of closed patterns in Table 3.3 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 4.1. For simplification of the presentation, the iterations before l_t = 2, i.e., for l_t = 5 and l_t = 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.

4.5 Selection Strategy

The Semi-MultiCons result for the support example, which execution is depicted in Table 4.1, is presented in the hierarchical graphical representation shown in Figure 4.3.

Duplicated consensus clustering solution are generated for iterations l_t = 4 and l_t = 2. This identical solution is then represented only once in the final hierarchical graphical representation with its associated count of how many times this consensus clustering occurs (not shown in the figure). This count represents the frequency of the consensus solution among all possible solutions, denoting its stability in the consensus generation process, and thus the robustness of its constituting clusters regarding the search space in input. Regarding the final hierarchical graphical representation for the support example shown in Figure 4.3, 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

Table 4.1: 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_t = 5 and 4 are identical to corresponding iterations in Figure 3.4. For l_t = 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 supervised 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 4.3.

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

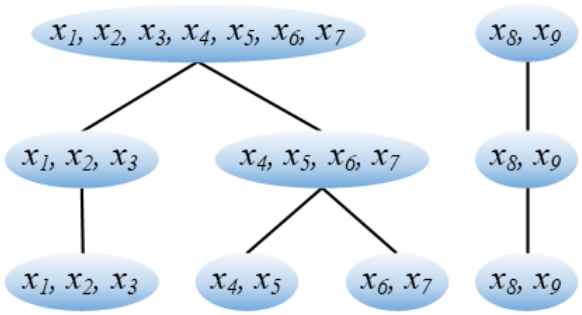


Figure 4.3: Hierarchical consensus clustering of the Semi-MultiCons approach. The three consensus clusterings generated from closed patterns and constraints are represented as a level in the hierarchical representation. The levels show the successive groupings of instances, each corresponding to a different 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.

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 4.3, 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.

4.6 Conclusion

In this chapter, we propose a new semi-supervised consensus clustering approach, named Semi-MultiCons. The major contribution of this work is the development of an iterative constraint-based merging/splitting consensus function based on MultiCons, and the optimization of the implementation based on the MultiCons approach to reduce computational complexity. In order to properly recommend the final result to user, we also propose a new constraint-based consensus selection method.

Chapter 5

Experimental Settings

In this chapter, we introduce our study datasets and experiment settings that were used for almost all the experiments in the thesis. The datasets consists are divided into two parts. The first part includes benchmark open-source clustering datasets, in order to compare our approach with other approaches in the state-of-art literature. The second part was created based on Amadeus data, to evaluate our method in the context of realistic use cases. Experiment settings include the choice of base clustering algorithms, the choice of input parameters, the strategy of constraint generation, the evaluation indexes, and so on.

5.1 Benchmark Datasets

Five classical benchmark datasets from the UCI Machine Learning Repository [26], namely the Iris, Wine, Seeds, Zoo and Ecoli datasets, are used to evaluate the Semi-MultiCons semi-supervised ensemble clustering performance using the standard Normalized Mutual Information index (NMI) [65]. Besides these datasets, the MNIST dataset [49], 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.

Some basic facts about these datasets are stated in Table 5.1. Both these six well-known benchmark datasets, and the evaluation indexes, presented hereafter 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.

Table 5.1: Benchmark dataset properties. For each of the six experimental datasets, the number of classes, the number of attributes and the number of instances are shown.

Dataset	Classes	Attribute	Number of instances
Iris	3	4	150
Wine	3	13	178
Seeds	3	7	210
Zoo	7	17	101
Ecoli	8	8	336
MNIST	10	784	70000

5.1.1 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 class. 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.

5.1.2 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 reported theoretical and experimental results in the literature, and availability and results of tests of implementations, the following semi-supervised clustering algorithmic approaches were selected: Third model (GV3) from [34], soft least squares Euclidean consensus (DWH) [24], hard Euclidean consensus (HE) [41] and metric pairwise constrained K-means (MPC-Kmeans) [8]. Implementations of these approaches can be found in R packages *clue* [40] and *conclust* [71].

Table 5.2: Input parameters for benchmark datasets. 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	k	Base clustering (unsupervised)	Base clustering (semi-supervised)
Iris	$[2-6]$	[Kmeans]	[MPC-Kmeans]
Wine	$[2-6]$	[Kmeans]	[MPC-Kmeans]
Zoo	$[5-9]$	[Kmeans]	[MPC-Kmeans]
Ecoli	$[4-8]$	[Kmeans]	[MPC-Kmeans]
MNIST	$[8-12]$	[Kmeans]	[MPC-Kmeans]

5.1.3 Input Parameters

For consensus clustering, base clustering algorithmic configurations must be defined to generate ensemble members. The range of values for the k parameter used for the ensemble member generation are shown in Table 5.2. In operational applications of clustering, when k is unknown, a commonly used idea is to estimate its value based on a small sample set of data. However, it is possible that the estimated k deviates from the number of clusters. The range of k values used for the experiments were defined as an approximation of the number of classes. For Amadeus datasets, the range of k is determined based on user preference about error tickets group size, with regard to correction process features.

Kmeans algorithm is selected as unsupervised base clustering approach because it is the most widely used and well-known clustering algorithm. The single semi-supervised clustering MPC-Kmeans approach, which is a semi-supervised variant of Kmeans, is used for both comparison with Semi-MultiCons semi-supervised consensus clustering approach, and to generate semi-supervised base clusterings in the clustering ensemble when comparing Semi-MultiCons with other semi-supervised consensus clustering approaches.

The MPC-Keans approach [8] was chosen since, to the best of our knowledge, it was reported in the literature as one of the most efficient single semi-supervised clustering approaches. Chapter 8 gives an extend analysis of other semi-supervised clustering algorithms regarding their performance and robustness in presence of noisy input. MPC-Kmeans is proved to be one of the most accurate and robust algorithms, regardless time complexity.

The number of constraints ranges from 0 to 210 for UCI Machine Learning Repository datasets.

For each number of constraints, 100 different constraint sets were generated to get repeated trials.

For the MNIST dataset, the potential negative effect of constraints (see Section 6.1.7 for details) was not studied due to the limitations of the MPC-Kmeans approach for such a large dataset. Indeed, for the negative effect experiment, repeated trials are necessary in order to get rid of potential randomness introduced by 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 6 000 to demonstrate the Semi-MultiCons ability to be applied on large and challenging, regarding the number of instance classes, datasets. Each consensus clustering approach was run ten times, and both Semi-MultiCons and other approaches are guaranteed to access exactly the same constraints for each run.

5.1.4 Evaluation Indexes

Normalized Mutual Information (NMI) [48], Clustering Accuracy (ACC) [87] and Purity [62] indexes are used to evaluate the quality of 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 result 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 clusters k_b that provides the best performance for the approach 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.

5.2 Amadeus Datasets

Data collected from the Revenue Accounting Workflow of Amadeus contain all accounting information required for processing a travel that is coded internally as a ticket in input of the workflow. A ticket is represented and modelled by using a Business Object Model (BOM), that is a hierarchical data structure representing the complete travel and its associated coupons, each coupon correspond-

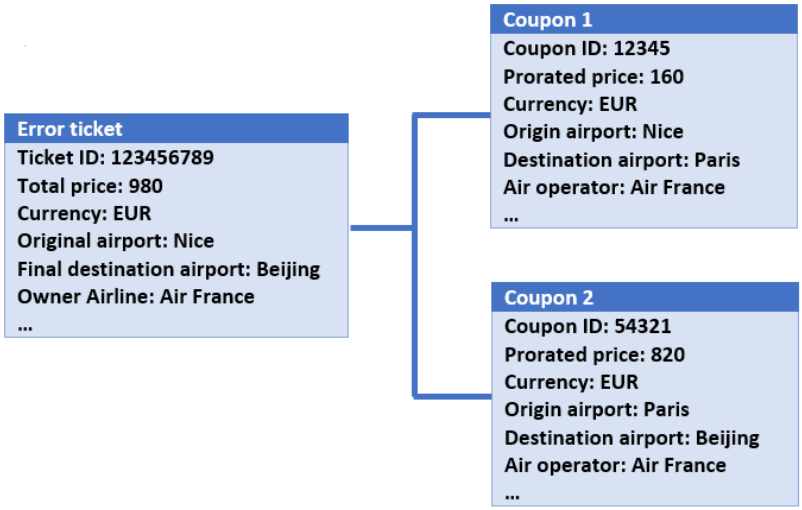


Figure 5.1: Example Business Object Model (BOM) for ticket. BOM is a hierarchical data structure representing the complete travel and its associated coupons, each coupon corresponding to a flight connection and related commercial treatments in the travel.

ing to a flight connection and related commercial treatments in the travel. For each ticket represented as a BOM, general data on the travel (total price, origin departure and final arrival airports, etc.) are included, as well as data on each coupon (departure and arrival airports, air operator, price, taxes, etc.). See Figure 5.1 for a concrete example of BOM.

Both the heterogeneity and number of features associated with each ticket present a great variability, depending on the module where it is raised. For example, if an error ticket is raised from Proration module, it means there is an error while amounts on ticket level are prorated to coupon level. Proration module only utilizes several data on ticket level, which largely restricts the number of features we should select from original data. As another example, consider an error ticket raised from the Interline module. Interline represents the process of negotiation between two airlines about the price of a coupon. In this case, only coupon level data is relevant with regards to the cause of error and correction process. We select Proration error tickets and Interline error tickets, from July 2019 to December 2019, to construct Amadeus dataset as they provide the largest volume of information in error ticket database. Also, each Amadeus airline customer has its individual database, and we selected the three customers which provide the largest volume of information. Some basic facts about the Amadeus datasets used during the experiments are stated in Table 5.3.

Table 5.3: Amadeus dataset properties. For each of the six industrial datasets, its number of estimated classes, number of attributes and number of instances are shown.

Datasets	Estimated Classes	Attribute	Number of instances
Proration 1	6269	10	10720
Proration 2	6273	10	21778
Proration 3	57923	10	101524
Interline 1	6964	20	39860
Interline 2	26446	20	92607
Interline 3	62121	20	121359

5.2.1 Pre-processing

Different pre-processing steps were tested in order to represent the information about error tickets in a relevant format regarding the applicability of unsupervised and supervised algorithms versus the heterogeneity, the number of objects and the number of variables in the processed datasets.

Most clustering algorithms require input attributes and output variables to be numeric and cannot be directly applied to categorical attributes. In the experiments, by using one hot encoding technique, categorical attributes are represented as binary attributes. Each categorical value is represented as a binary attribute, and categorical attributes are encoded as vectors with all '0' values except for the index of the categorical attribute value, which is marked as '1'.

Scaling is another essential step for clustering algorithms as they calculate distances between data. If not scale, the attributes with a larger value range will dominate when calculating distances. In the experiments, numerical attributes are normalized to range [0, 1] by using min-max scaler.

5.2.2 Constraint Generation

For benchmark datasets, the ground truth class is available for each instance. However, for Amadeus datasets, we can only access user corrections on error tickets. Therefore, class label estimation is essential. We applied a strict policy on user corrections to generate class label: Two corrections are assigned a same class label only if they are exactly the same. With this strict policy, it is more likely to have more classes than ground truth and there is a large chance that two instances with different class labels are still similar. Thus in Table 5.3, the number of classes is large. The must-link and cannot-link constraints are generated based on the estimated classes, using the same

Table 5.4: Input parameters for Amadeus datasets. For each of the Amadeus datasets, the range of values for the k parameter defining the number of clusters extracted in the base clusterings are shown.

Dataset	Batch size	Number of batches	k for a batch	Base clustering (unsupervised)
Proration 1	5000	3	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]
Proration 2	5000	5	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]
Proration 3	5000	21	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]
Interline 1	5000	8	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]
Interline 2	5000	19	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]
Interline 3	5000	25	$[2, 50, 200, 250, 1000, 2500]$	[Kmeans]

strategy as for UCI MLR benchmark datasets.

5.2.3 Input Parameters

Considering the complexity of single semi-supervised clustering algorithms and the volume of Amadeus datasets, Semi-MultiCons with semi-supervised base clustering was not applied on Amadeus datasets since ensemble generation step is extremely time consuming. Instead, the same unsupervised base clustering algorithm as tested for benchmark datasets, that is the Kmeans algorithm, was selected. Considering functional requirements and industrial scenarios, the datasets listed in Table 5.3 were divided into mini-batches, based on creation timestamp of data instances, to speed up the clustering process. The final clustering was generated by appending results of the mini-batches. The size of batch and the range of values for the k parameter, shown in Table 5.4, were determined by Amadeus Revenue Accounting experts based on their domain knowledge.

Due to the size of the Amadeus datasets, the number of constraints was fixed to 100 per batch and each consensus approach was run 10 times.

5.2.4 Evaluation Index

We used Purity [81] as an important evaluation index, but not ACC for Amadeus datasets. Purity is a measure of how well a cluster contains a single class [62]. This is an important index in industrial scenarios because error ticket cluster proposed by our approaches is assumed to be reliable and

accurate, on which similar correction actions will be applied. An increment id i is assigned to each level in the hierarchical consensus clustering of the Semi-MultiCons, from the bottom to the top, starting with 0. The NMI index score, Purity index score and the inferred number of clusters k of the level i are noted as S_i, P_i and k_i respectively. The performance of other consensus clustering approaches is not demonstrated, since they require k as an input parameter. For Amadeus datasets, the clustering result presented to the user is always the lowest level in the generated hierarchy, while the user is able to explore upper levels if needed. See Chapter 7 for more details.

Chapter 6

Experimental Results

We report the experimental results of the Semi-MultiCons approach on the benchmark and Amadeus datasets in this chapter. As discussed in Chapter 4, supervised information represented as constraints can be integrated in different phases of Semi-MultiCons. To completely evaluate the effect of constraints, we implemented four different variants of the Semi-MultiCons approach:

- MultiCons (MC): Original MultiCons approach.
- Semi-MultiCons (SMC): MultiCons with constraint-based merging/splitting consensus function and constraint-based consensus selection.
- MultiCons with semi-supervised base clustering (MC-s): Original MultiCons but unsupervised base clusterings are replaced with semi-supervised base clusterings.
- Complete Semi-MultiCons (cSMC): Semi-MultiCons but unsupervised base clusterings are replaced with semi-supervised base clusterings.

As explained in the previous chapter, MC and SMC approaches utilize Kmeans as base clustering approach while MC-s and cSMC approaches utilize MPC-Kmeans. These variants of the Semi-MultiCons approach are evaluated and analyzed in this chapter. In the following part, Semi-MultiCons generally refers to all the Semi-MultiCons approaches used in the experiment, while SMC specifically refers to the second approach in the previous list.

6.1 Performance of Semi-MultiCons on Benchmark Datasets

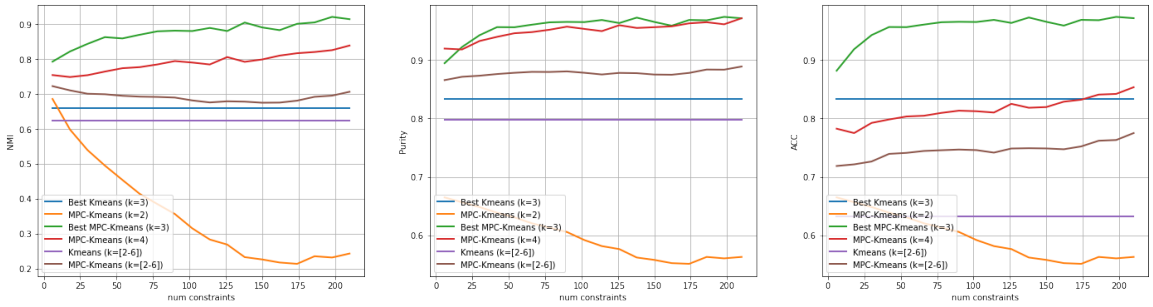
An important difficulty in the application of most clustering and consensus clustering methods is the setting of the k parameter, that defines the number of clusters that will be generated. Indeed, an incorrect value for the k parameter may lead to a significant decrease in the relevance of the clustering result. Another difficulty, related to the usage of supervised information in semi-supervised clustering, 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 [21, 77, 100]. During the experiments of the Semi-MultiCons approach on the benchmark datasets, 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 Semi-MultiCons and other semi-supervised clustering and/or consensus clustering approaches.
- Assessment of the potential negative effect of integrating constraints in the Semi-MultiCons process.
- Scalability and complexity of the Semi-MultiCons approach for processing very large datasets.

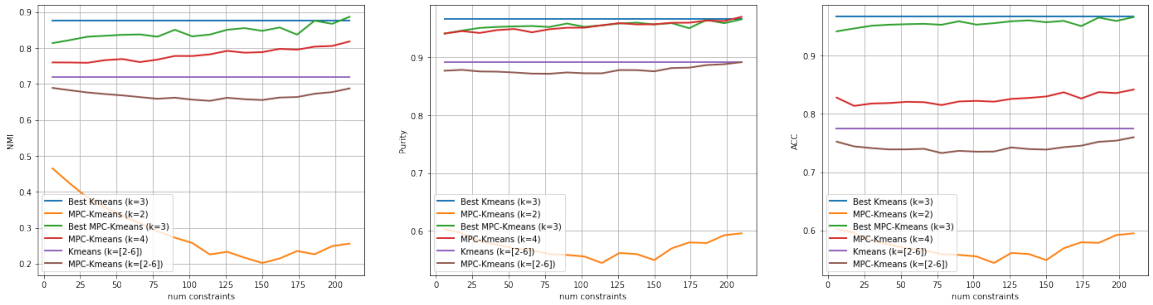
The experimental settings, detailed in Chapter 5, were defined to address more specifically these central questions.

6.1.1 Performance of Base Clustering Approaches

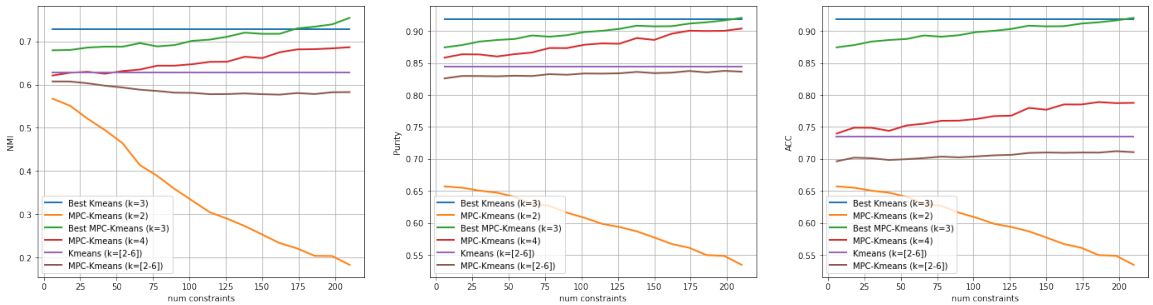
Figure 6.1 demonstrates the performance of base clustering algorithmic approaches, Kmeans and MPC-Kmeans, on the Iris, Wine, Seeds, Zoo and Ecoli datasets. The blue curve represents the best performance that can be achieved by Kmeans and its corresponding number of clusters k among all tested values in 5.2. The three curves depicting the evaluation of MPC-Kmeans results correspond each to a different value for the input parameter k. The green curve corresponds to the k_b value number of clusters, that generates the best result among all tested values. The orange curve corresponds to the $k_b - 1$ value, and the red curve corresponds to the $k_b + 1$ value. The purple and brown curves represent the average evaluation index score (NMI, Purity or ACC) of Kmeans and MPC-Kmeans results, respectively, over all tested number of clusters.



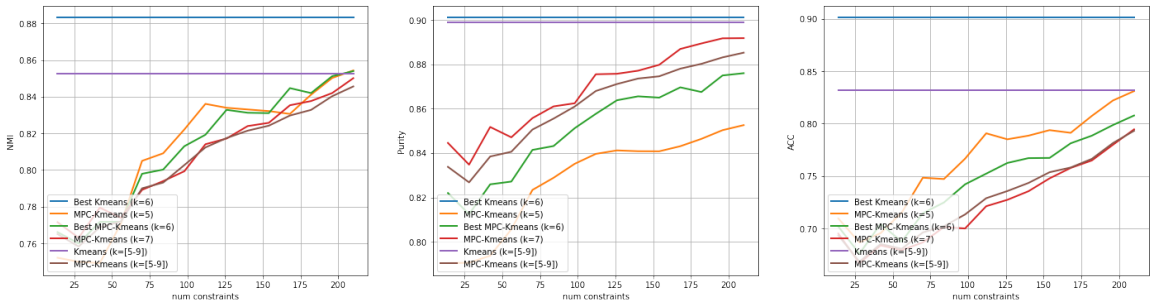
(a) Iris



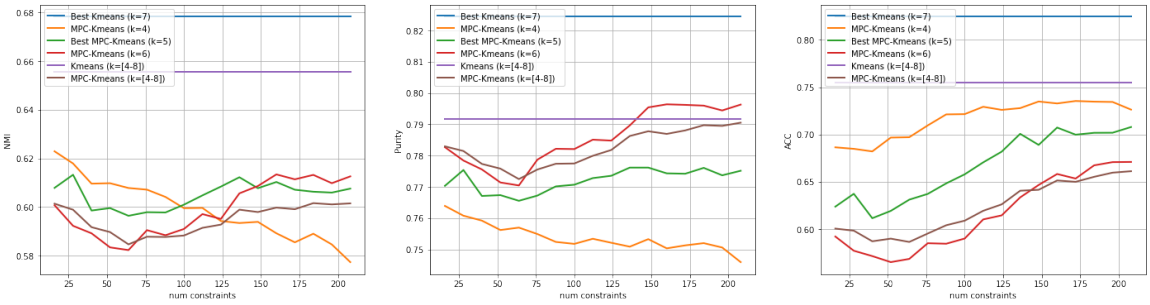
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.1: Performance of base clustering approaches. The curves illustrate the best and the average clustering results obtained by the MCP-Kmeans single clustering approach and the Kmeans single clustering approach for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the evaluation index value (NMI, Purity or ACC) of the clustering solution generated.

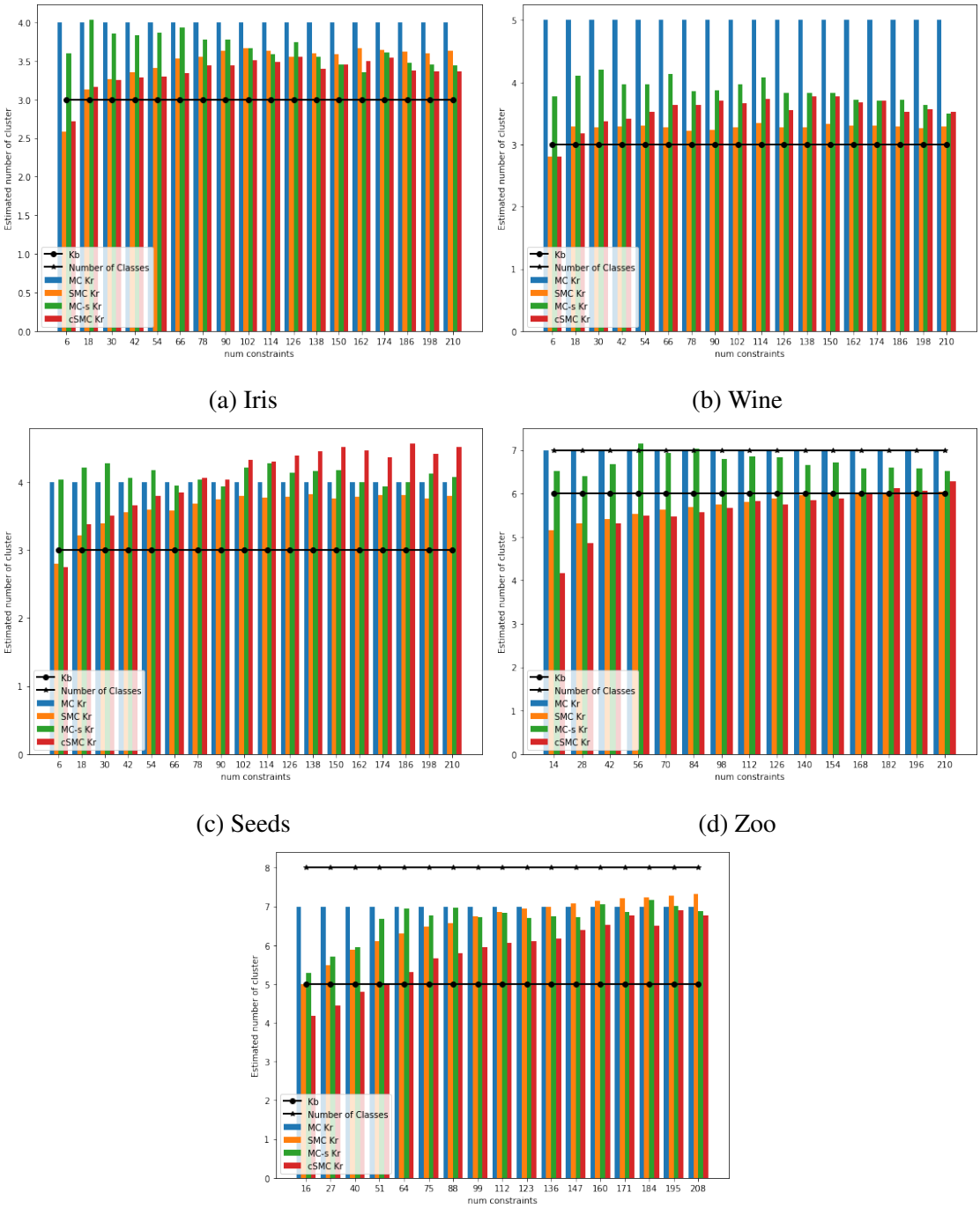
Considering the MPC-Kmeans approach, we observe that its performance strongly depends on the value of its input parameter k. If this value does not fit well with the number of clusters in the dataset, MPC-Kmeans fails to give good performance. Also, for imbalanced Zoo and Ecoli dataset, MPC-Kmeans achieves 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. We also observe that, sometimes the best performance of MPC-Kmeans is worse than unsupervised Kmeans, especially for Zoo and Ecoli datasets, showing the potential negative effect of integrating constraints.

Considering the Kmeans approach, it deviates the number of classes as well for Zoo and Ecoli dataset. Moreover, it achieves its best performance when k equals to 2 for Iris dataset, corresponding to a clustering result that blindly assign instances from two classes to a same cluster.

This experiment shows the importance of the k parameter setting for Kmeans and MPC-Kmeans approaches.

6.1.2 Comparison between Inferred and Real Number of Classes

The average inferred numbers of clusters k_r by Semi-MultiCons approaches over all trials per number of constraints for each dataset are presented in Figure 6.2. k_b represents the number of



(a) Iris

(b) Wine

(c) Seeds

(d) Zoo

(e) Ecoli

Figure 6.2: Comparison between the inferred and ground truth numbers of classes. For each of the five datasets, the average number of clusters k_r inferred by Semi-MultiCons approaches, the number of clusters k_b for MPC-Kmeans to generate the best base clustering solution, and the true number of classes in the dataset are shown.

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 6.1.4. The real, i.e., ground truth, number of classes in the dataset is also given as reference. We can 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. With the number of constraints increases, the inferred k_r provided by the different Semi-MultiCons approaches come closer to each other, as shown by the orange, green and red bar, implying that the approaches reach an agreement about the number of clusters in dataset.

For the Zoo and Ecoli datasets, Semi-MultiCons infers a smaller k_r than the number of classes, corresponding to the fact that these datasets actually have classes that only contain few instances, even less than 5% of the number of total instances. The Zoo dataset has two classes that contain only 4 and 5 instances, and the Ecoli dataset has three classes that consist of only 2, 2 and 5 instances. For Iris, Wine and Seeds datasets, Semi-MultiCons 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.

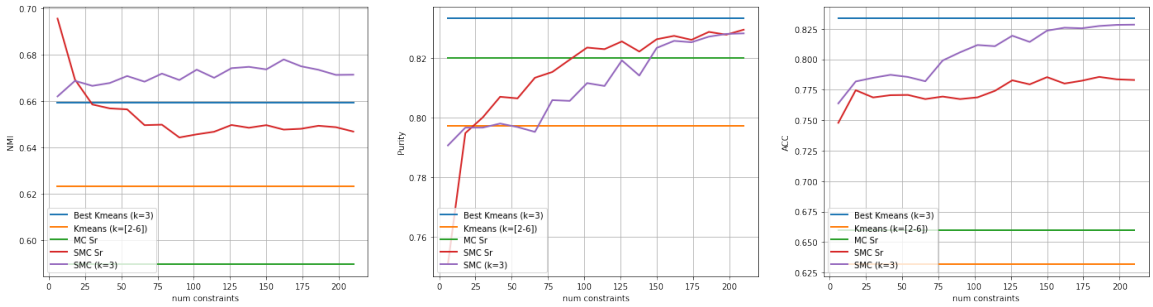
Actually, several sub-spaces in the data space defined by the input dataset, that is intrinsic clusters in this data space, can correspond to the same class. This is the case if the class corresponds to different subgroups of instances, which means the class can be characterized by several distinct groups 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.

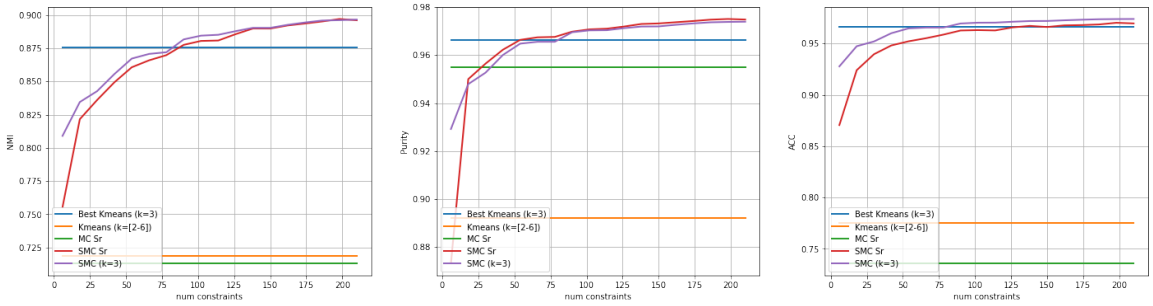
The multi-level structure of clusters in output of Semi-MultiCons can provide information about this property, by showing the successive merging and splitting operations performed, and help to automatically discover the appropriate number of clusters k for the dataset, that is to identify the most relevant consensus, i.e., level in the hierarchy, among the consensuses in output.

6.1.3 Comparison with Single Unsupervised Clustering Approaches

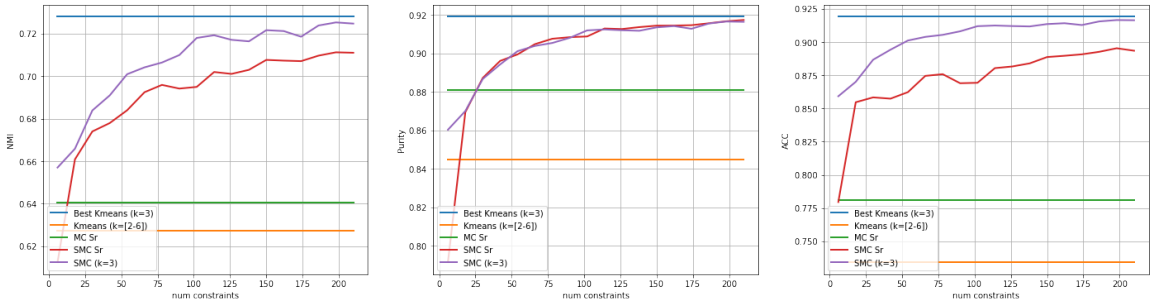
The results of the Semi-MultiCons approaches and Kmeans single unsupervised clustering approach are compared in this experiment. In particular, MC and SMC approaches are selected as



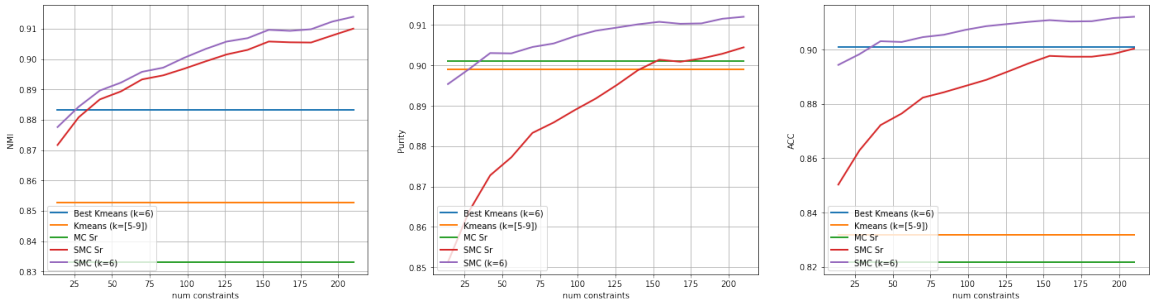
(a) Iris



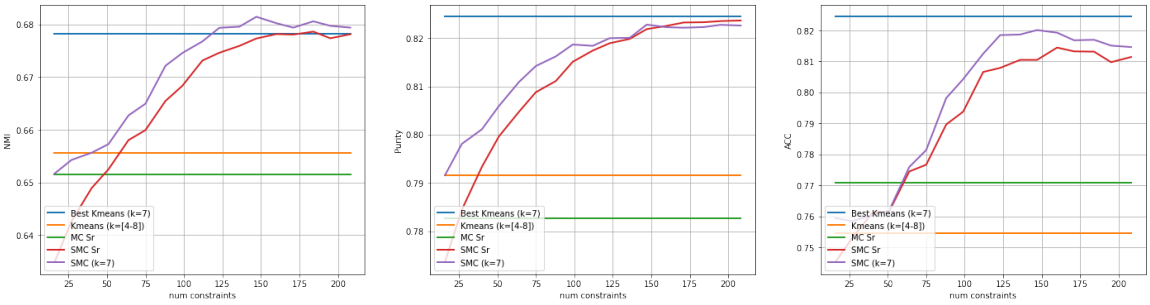
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.3: Comparison between Semi-MultiCons and Kmeans approaches. The curves illustrate the best clustering result obtained by the Kmeans single clustering approach and the Semi-MultiCons consensus clustering approach for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the evaluation index value (NMI, Purity or ACC) of the clustering solution generated.

their consensus results are generated based on Kmeans.

As stated before, three indexes, namely NMI, Purity and ACC, are used to evaluate and compare the relevance of the generated clustering solutions, and the number of pairwise constraints used was varied between 0 and 210 to evaluate their impact on the clustering result.

The blue curve represents the best performance that can be achieved by Kmeans and its corresponding number of clusters k among all tested values in 5.2. The orange curve represents the average evaluation index score of Kmeans results over all tested number of clusters. The three curves depicting the evaluation of Semi-MultiCons results correspond each to a different approach or number of clusters. The green curve and red curve correspond to the S_r evaluation in the situation where the number of classes is unknown for MC and SMC, respectively, that is the consensus clustering evaluated is the one automatically selected by the approach in the output hierarchy. The purple curve represents the S_r evaluation when the number of clusters k is given to SMC as input, that is the consensus clustering evaluated is the one with a number of clusters equal to k in the output hierarchy.

The baselines are the average performance of Kmeans and the result of the MC approach, as the constraints are not integrated into these two approaches. We can clearly see from the curves that the SMC approach outperforms the baselines and its performance increases with the number

of constraints, proving that constraints are useful for improving the quality of the clustering. An exceptional case is Iris dataset. The NMI index score of SMC drops when the number of constraints increases. A potential cause is that Kmeans achieves its best performance when the number of clusters k corresponds to two, which differs from the ground truth number of classes, and SMC tends to approach this number when the number of constraints increases. The degraded performance is therefore a trade off at the detriment of the accuracy of the inferred number of clusters.

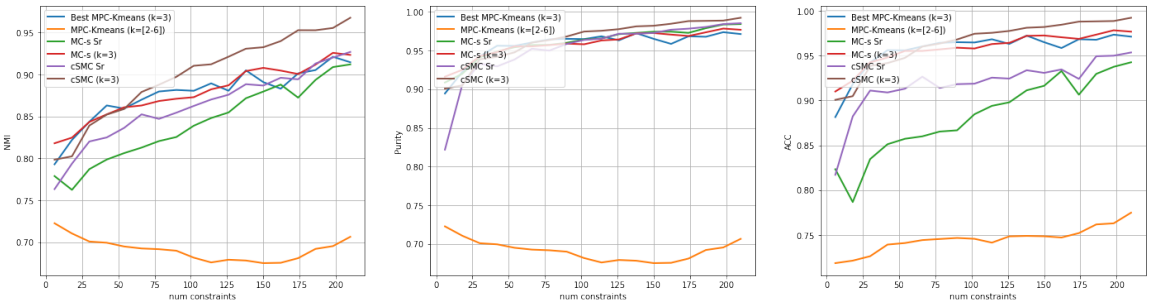
We can also see that, without knowing the number of clusters, SMC is able to provide good performance, close to the best performance achieved by Kmeans, as shown by the red curves. When the number of clusters k, corresponding to the number of clusters which produces the best Kmeans performance, is given, as shown by the purple curve, the SMC approach is able to give an even better performance.

6.1.4 Comparison with Single Semi-supervised Clustering Approaches

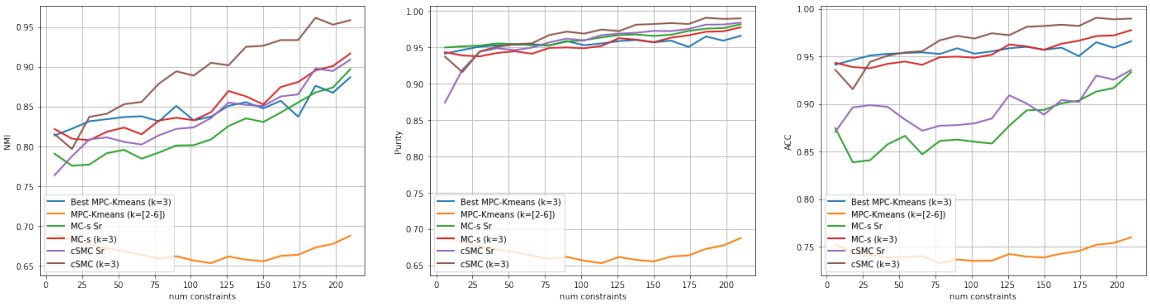
In this experiment, the results of the Semi-MultiCons approaches and MPC-Kmeans single clustering approaches are compared in terms of relevance of the consensus solution generated. The results of MC-s, cSMC and MPC-Kmeans for the Iris, Wine, Seeds, Zoo and Ecoli datasets are presented in Figure 6.4. The average performance obtained for all trials of each run is shown. The MC approach and SMC approach are excluded from this experiment since their consensus results are generated based on Kmeans instead of MPC-Kmeans. 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 6.1.2.

The blue curve corresponds to the k_b number of clusters, that generates the best MPC-Kmeans result among all tested values in 5.2. The orange curve corresponds to the average evaluation index scores of MPC-Kmeans over all tested number of clusters. The green and purple curves represent the S_r evaluation in the situation where the number of clusters is unknown for MC-s and cSMC respectively. The red and brown curves represent the S_r evaluation when the number of clusters k_b is given to MC-s and cSMC, respectively, as input, that is the consensus clustering evaluated is the one with a number of clusters equal to k_b in the output hierarchy.

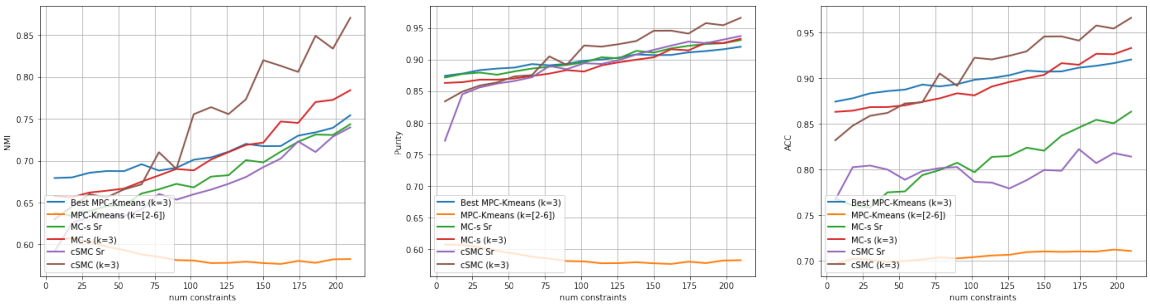
Both MC-s and cSMC have better performance than the average evaluation index scores of MPC-Kmeans over all tested number of clusters, proving the positive effect of the consensus process of the Semi-MultiCons approaches. The curves also illustrate that the performance of MC-s and cSMC



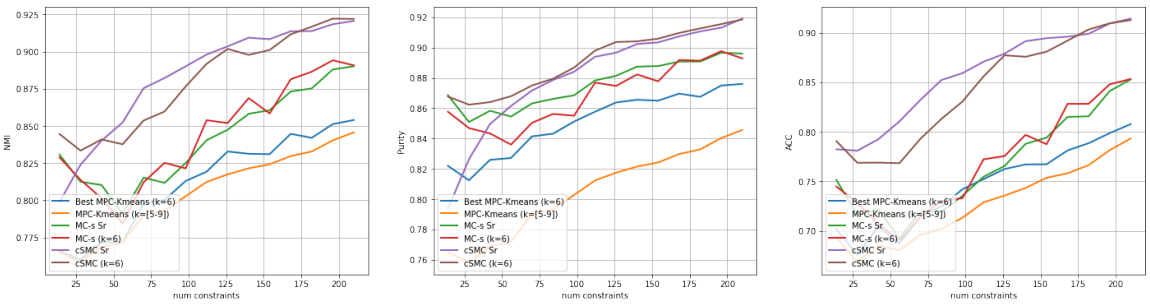
(a) Iris



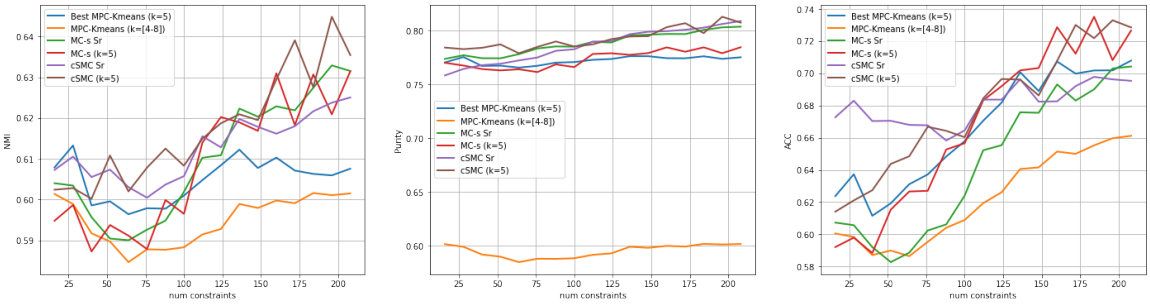
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.4: 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 approaches for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the evaluation index value (NMI, Purity or ACC) of the clustering solution generated.

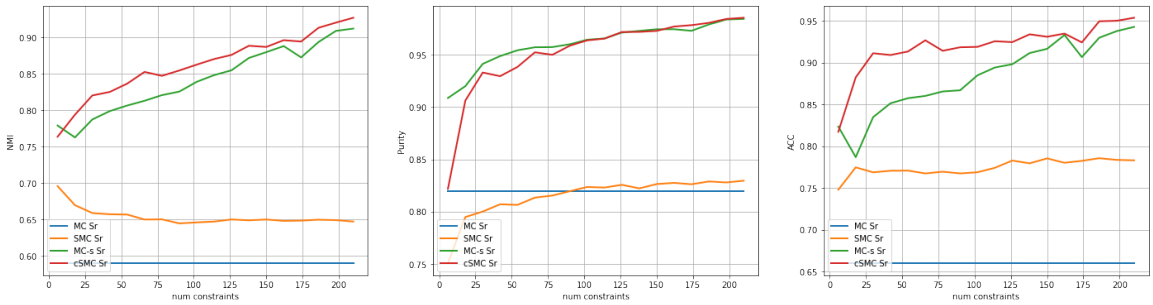
are improved with the number of constraints, showing the benefit of integrating constraints.

We can also find that MC-s and cSMC approaches are able to give a comparable or better performance when the number of clusters k_b is given as input, as shown by the red and brown curves, for all three evaluation indexes. Moreover, without knowing k_b, they are still able to give good performance, close to those obtained with known k_b, as shown by the green and purple curves, especially when the number of classes is large.

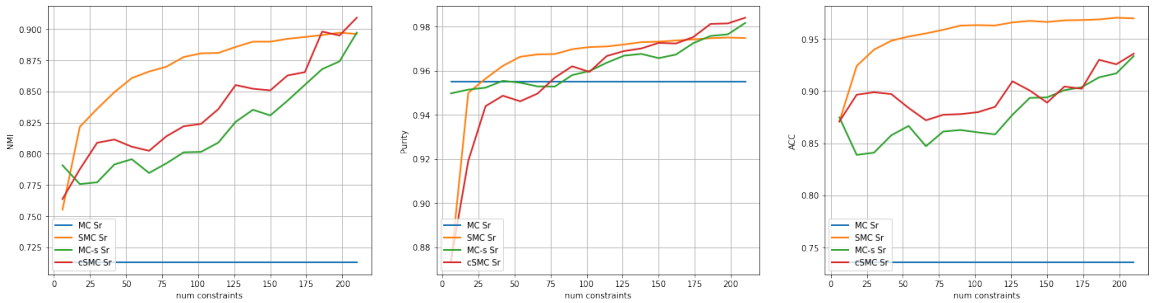
6.1.5 Comparison between the Semi-MultiCons Approaches

The results of the four Semi-MultiCons approaches are illustrated in Figure 6.5. The curve with different color corresponds to the S_r evaluation in the situation where the number of clusters is unknown for each Semi-MultiCons approach. The MC approach without using constraints, as shown by the blue curve, is represented as baseline.

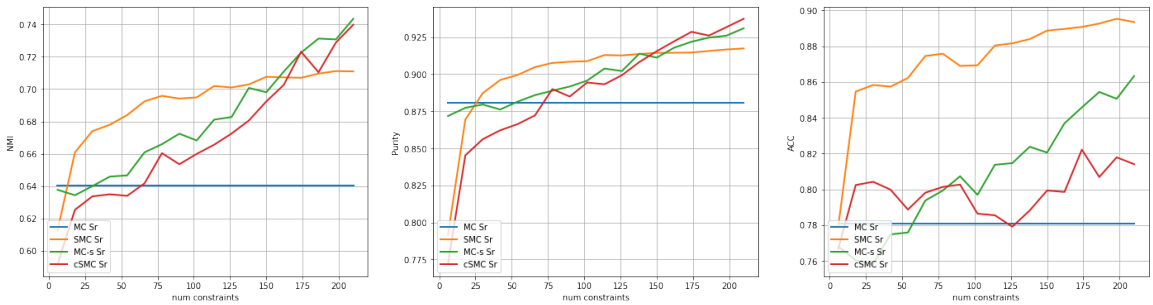
Overall, SMC, MC-s and cSMC approaches outperform the MC approach, except for Ecoli dataset, for which MPC-Kmeans performs extremely bad and therefore the performance of MC-s and cSMC are influenced. The performance of SMC sometimes does not grow anymore after reaching a certain number of constraints, as presented by the orange curve. In contrast, the red curve and green curve always increase with the number of constraints, displaying the potential of



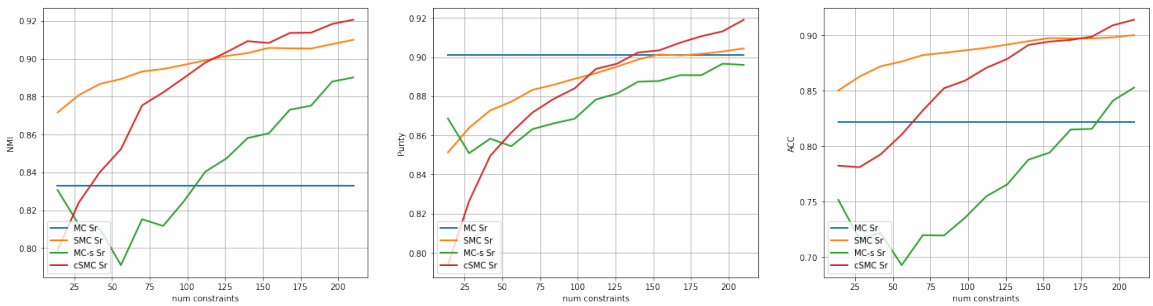
(a) Iris



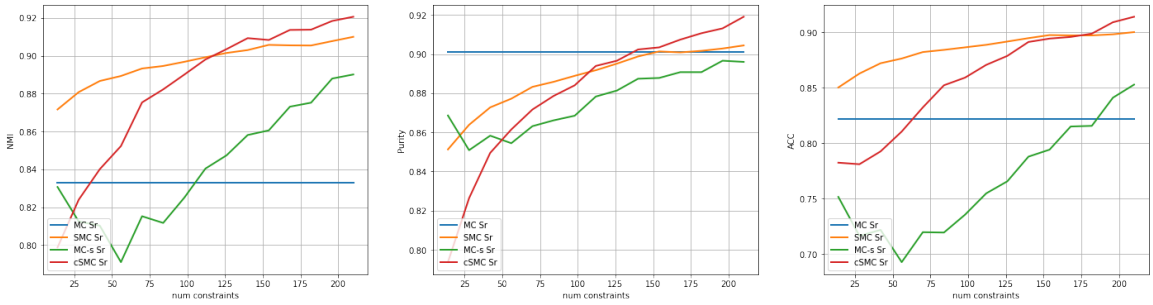
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.5: Comparison of the four Semi-MultiCons approaches. The curves illustrate the clustering result obtained by the MC, SMC, MC-s and cSMC approaches for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the evaluation index value (NMI, Purity or ACC) of the clustering solution generated.

the MC-s and cSMC approaches.

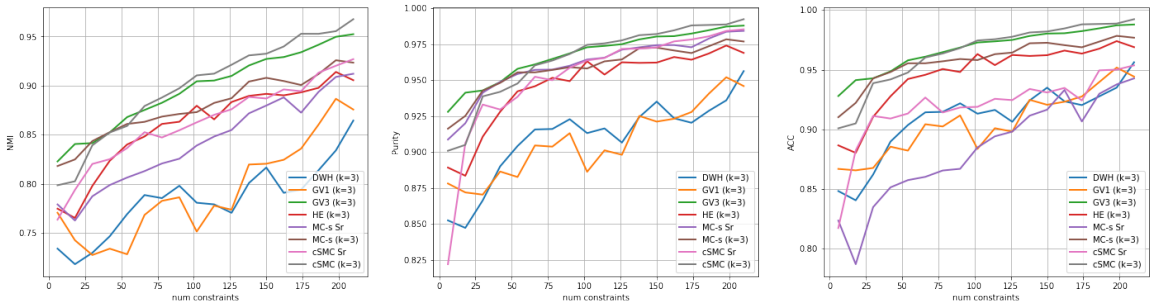
The difference between the red curve and the green curve illustrates the effect of our constraint-based merging/splitting consensus function since the only difference between them is that the cSMC approach further involves constraints into the consensus process. In most cases, this effect is positive.

The choice between Semi-MultiCons approaches mainly depends on the accessibility of base clustering approaches, the requirements on scalability and efficiency, etc. In experimental context, the cSMC approach is always selected as it fully integrates constraints in both ensemble member generation and consensus process. In real industrial scenarios, the SMC approach is preferred because generating base clustering results from semi-supervised clustering approaches can be quite time consuming.

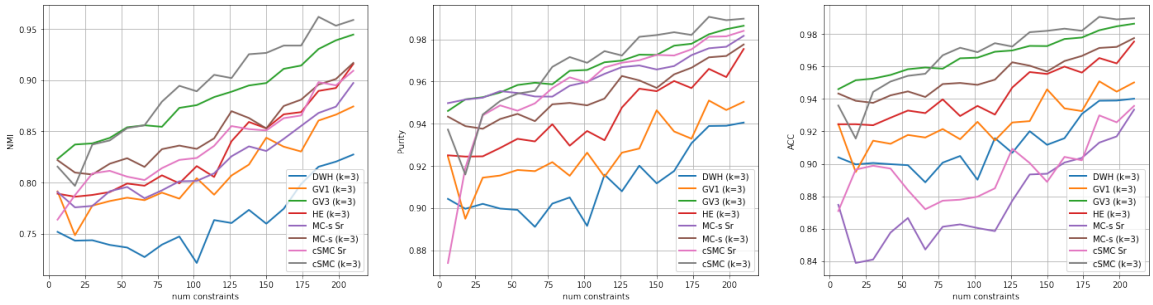
6.1.6 Comparison with Semi-supervised Consensus Clustering Approaches

This experiment compares the results of the Semi-MultiCons and other state-of-the-art semi-supervised consensus clustering approaches in terms of relevance of the consensus solution generated.

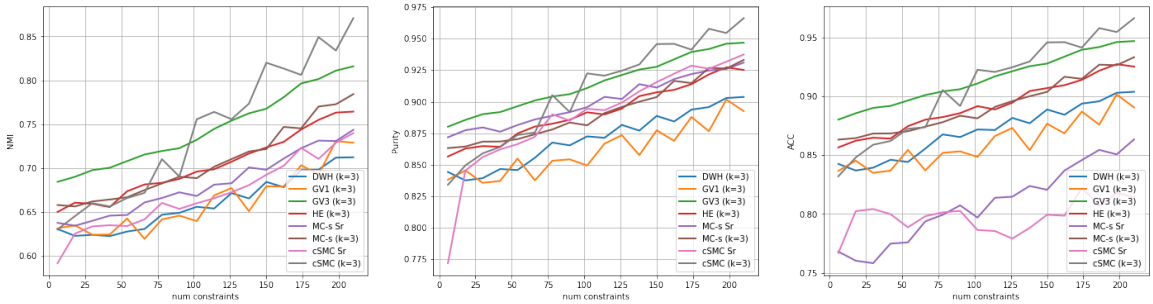
Experimental results for the Semi-MultiCons, DWH, GV3 and HE approaches for the Iris, Wine, Seeds, Zoo and Ecoli datasets are presented in Figure 6.6. The height of the curves represents the



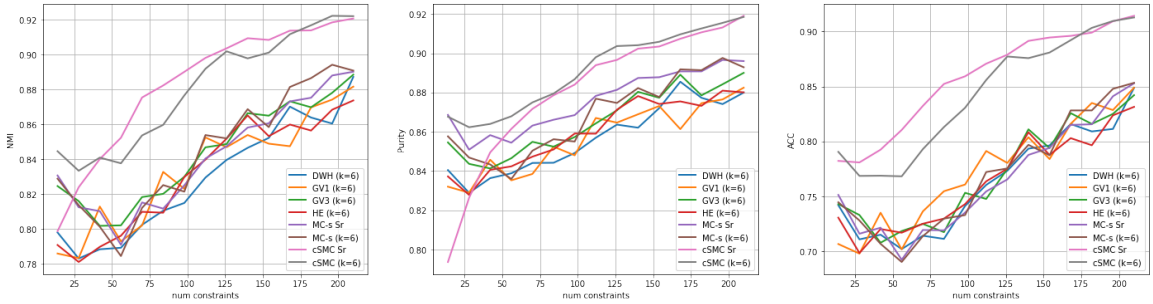
(a) Iris



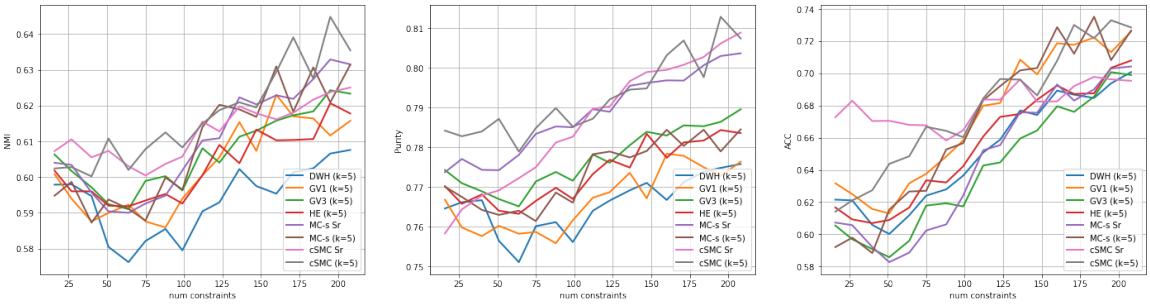
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.6: Comparison between Semi-MultiCons and other semi-supervised consensus clustering approaches. The curves illustrate the clustering result obtained by the DWH, GV1, GV3, HE and Semi-MultiCons approaches for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the evaluation index value (NMI, Purity or ACC) of the clustering solution generated.

value of the evaluation index for the clustering generated when varying the number of pairwise constraints between 0 and 210 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. The MC and SMC approaches are therefore not demonstrated as their base clusterings are different.

Obviously, 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. 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 four other approaches, results are similar for the five datasets: The best solutions are generated by the GV3 algorithm, the lower performer solutions are generated by the DWH algorithm, and the HE and GV1 algorithms generate 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.

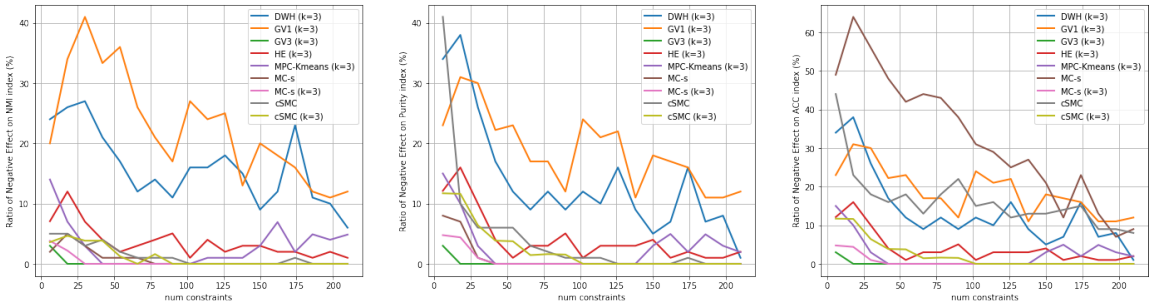
6.1.7 Analysis about Negative Effect

A potential *negative effect* issue of semi-supervised clustering methods was largely reported in the literature [21, 77, 100]. This issue relates to the use of pairwise constraints as supervised 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.

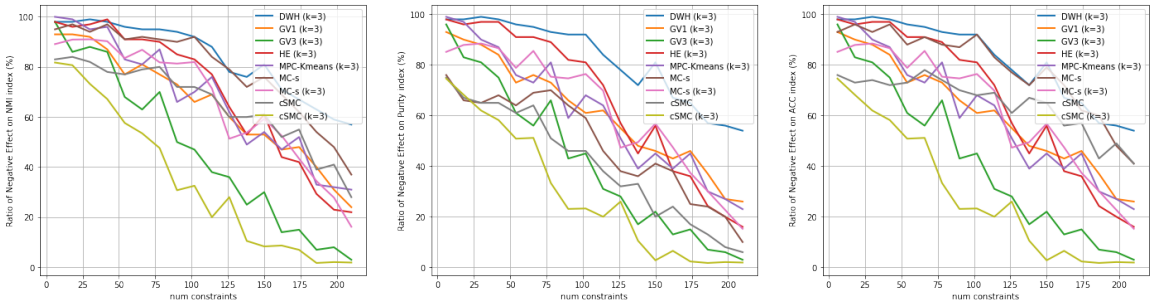
This experiment analyzes the ratio of negative effect occurrence for the Semi-MultiCons approach compared to other baseline semi-supervised clustering approaches. The importance of the negative effect of using constraints is evaluated by the fraction of times that unconstrained version produced better results than the constrained version. 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 given represent the proportion of occurrences that 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 6.7. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the fraction of times that the algorithm produced worse performance than the baseline Kmeans clustering with k_b as input parameter. The pink and yellow curves represent the evaluation of the MC-s and cSMC consensus solutions, respectively, when the number of clusters is equal to the optimal number of clusters k_b. The brown and gray curves show the negative effect ratio of the S_r evaluation in the situation where the number of clusters is unknown for MC-s and cSMC respectively. The five remained curves represent the evaluation of the semi-supervised results of the DWH, GV1, GV3, HE and MPC-Kmeans algorithms with a k input parameter equals to k_b.

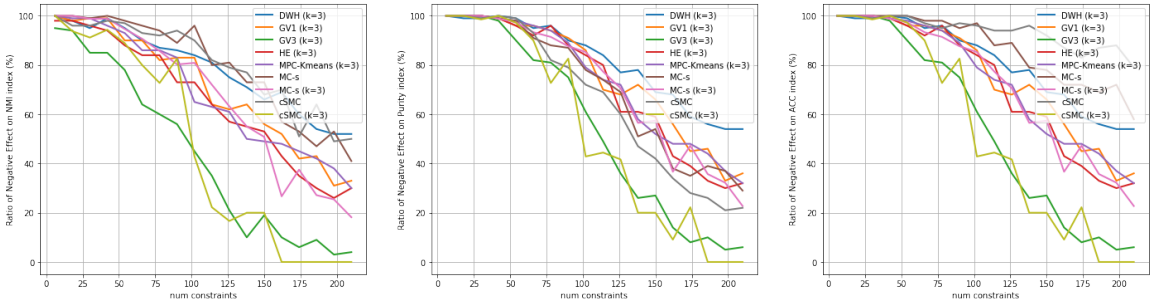
We can see from the results in the figure that for all algorithms, the negative effect is decreased with the number of constraints increases, implying that extending the size of constraint set can be a possible solution to fight against negative effect. Under the same condition where the optimal number of clusters is provided, cSMC highly reduce the occurrences of negative effect for most datasets, compared to other approaches, as illustrated by the yellow curve. When the optimal num-



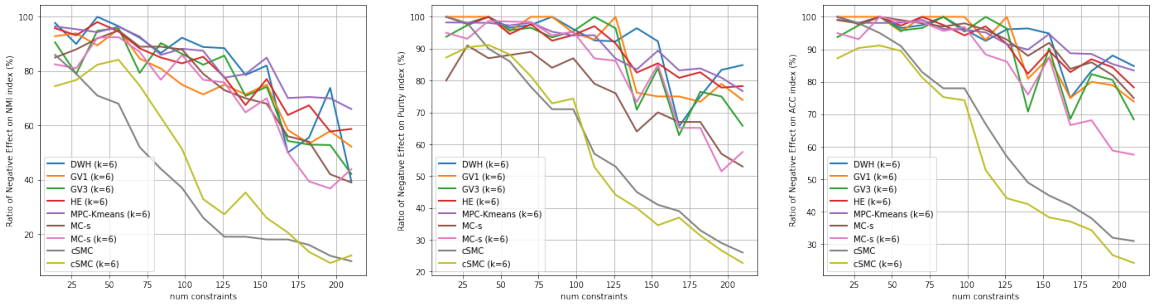
(a) Iris



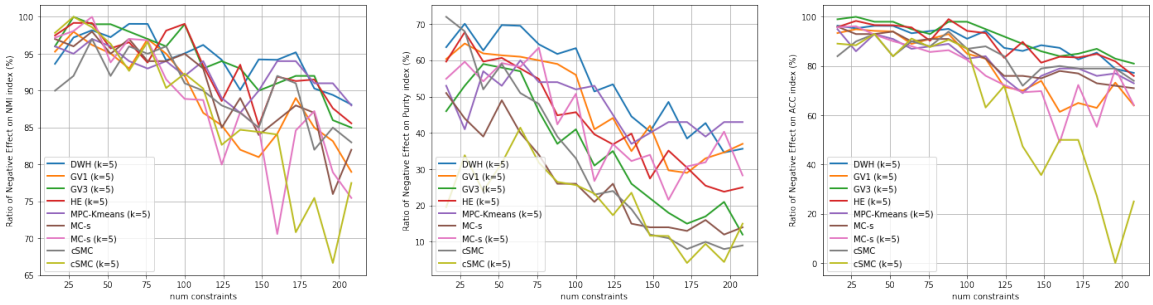
(b) Wine



(c) Seeds



(d) Zoo



(e) Ecoli

Figure 6.7: 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 Kmeans clustering with optimal number of clusters as input parameter. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the ratio of negative effect.

ber of clusters is not provided, Semi-MultiCons alleviates negative effect as well, especially for the cSMC approach which integrates constraints in both base clusterings generation stage and consensus process. For the Iris dataset, Semi-MultiCons and the GV3 algorithm eventually reach 0% negative effect when the number of constraints is large enough. 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 of each class.

6.1.8 Performance on the MNIST Benchmark Dataset

This experiment aims to compare the performances, in terms of execution times and applicability regarding memory usage, of the different semi-supervised consensus clustering approaches, based on MPC-Kmeans base clusterings. The MC approach and SMC approach are therefore not demonstrated as their base clusterings are different. 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 6.1. These results present both the average quality of the clustering in output and the execution times for each approach. The GV3 algorithm runs out of memory for such a large dataset since it requires 36.5 GB memory. Thus, this dataset exceeding its capacity regarding memory usage, its performances are not presented. Note that for this experiment, the optimal number of clusters is not

Table 6.1: 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 index and execution times in seconds.

Algorithm	NMI index	Purity index	ACC index	Time (s)
MC-s	0.7687	0.7566	0.7109	26.37
cSMC	**0.8081**	**0.8047**	**0.7893**	271.53
MPC-Kmeans (k=10)	0.7518	0.7207	0.6758	39006.80
DWH (k=10)	0.7547	0.7327	0.6935	**0.23**
HE (k=10)	0.7675	0.7424	0.6989	1.08
GV1 (k=10)	0.7581	0.7306	0.6904	3.53

provided to Semi-MultiCons.

We can see that among the six compared approaches, the DWH and HE approaches have the lowest execution times. However, as observed before, their performance is clearly lower compared to Semi-MultiCons and GV3 for datasets from 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 [72] of the clustering results for the compared approaches are demonstrated in Figure 6.8. The implementation in Scikit-learn Python package [55] was used, with initialization method set to PCA [64] and parameter perplexity set to 40, that is the same initialization setting as [72]. 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 6.1.

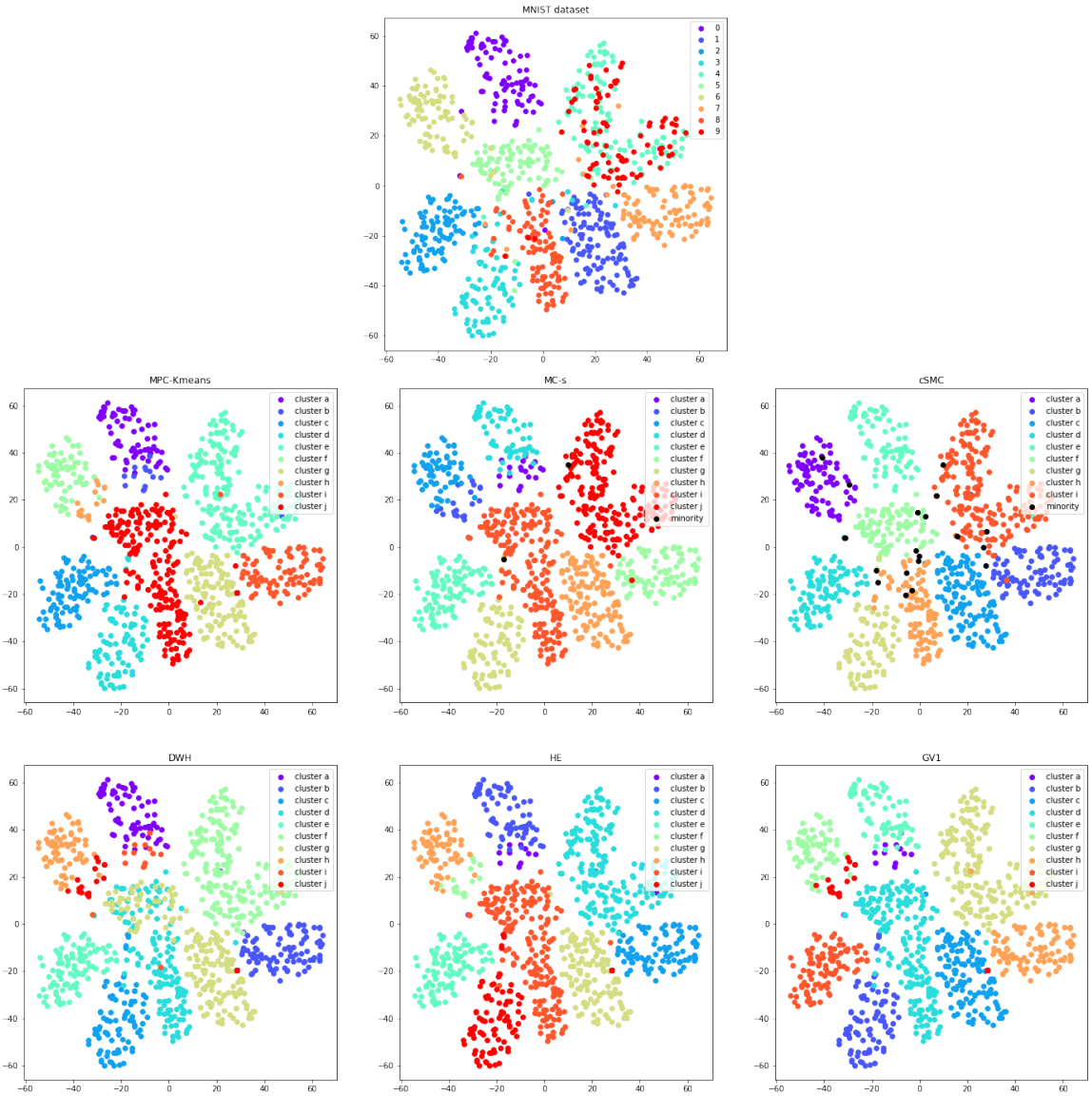


Figure 6.8: Representations of the clustering results of the semi-supervised single and consensus clustering approaches on the MNIST dataset of 70 000 instances using the t-SNE visualization. Figures show the clustering results for 1 000 sample instances in the latent space. The horizontal and vertical axis represent the latent space and the colors represent the clustering results.

6.1.9 Analysis about Convergence

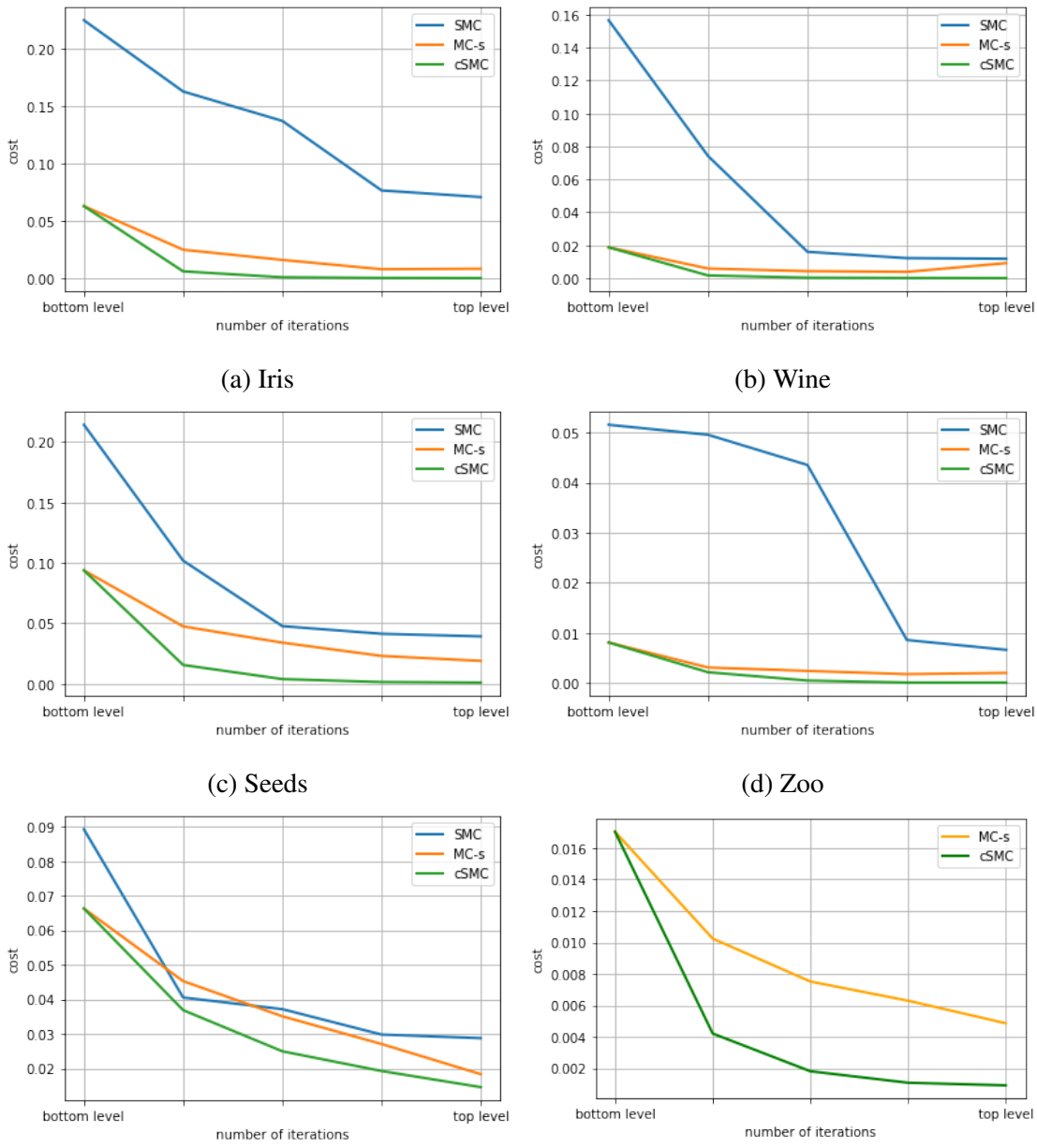
Since the proposed constraint-based consensus process is a dynamic process with merging/splitting operations, we evaluate the convergence of Semi-MultiCons in this experiment, to illustrate the trend of the cost regards to the number of iterations. The objective of the novel constraints-based consensus function, as explained in 4.4, 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 to 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 3.4 and Figure 4.1.

Results presented in Figure 6.9 show that for all the six 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 approaches. Among different approaches, the cSMC approaches has the best convergence as it integrates constraints in both the base clusterings and the consensus function.

6.1.10 Computational Complexity Study

This study aims to evaluate the efficiency of the Semi-MultiCons approach and compare it with other semi-supervised clustering approaches. During the first experiment of this study, the execution times of Semi-MultiCons and the five other semi-supervised clustering approaches used in the experiments are compared. Experimental results are presented in Figure 6.10. 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 eight approaches compared while varying the number of pairwise constraints used during the run. We can see that DWH, HE and GV1 are the most efficient approaches for the five 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 Semi-MultiCons execution times are systematically lower than those of MPC-Kmeans and GV3 for all the five datasets.

During the second experiment of this study, Semi-MultiCons is applied to different samples of the MNIST dataset. Only the cSMC approach is illustrated as it has the most important execution time among all four Semi-MultiCons approaches according to the result discussed in the previous section. Seven samples, containing from 10 000 instances for the smallest to 70 000 instances for



(a) Iris

(b) Wine

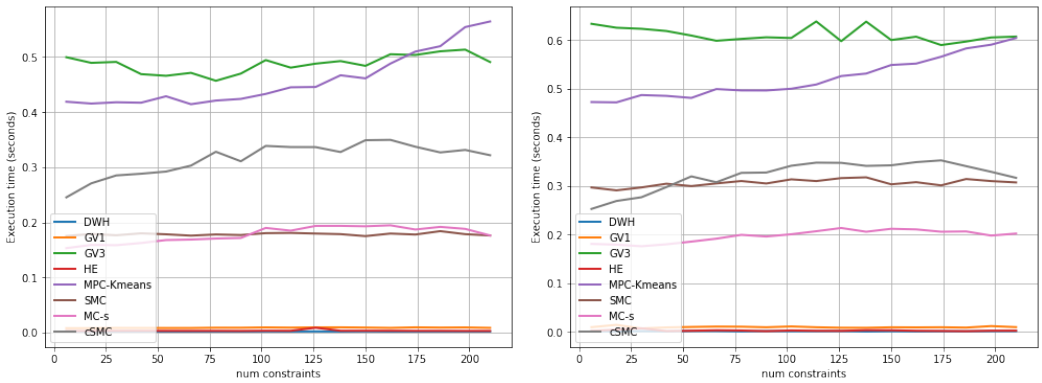
(c) Seeds

(d) Zoo

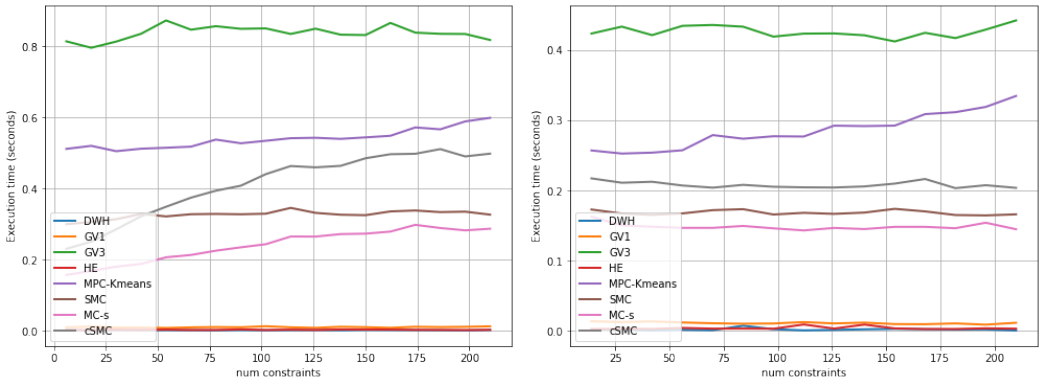
(e) Ecoli

(f) MNIST

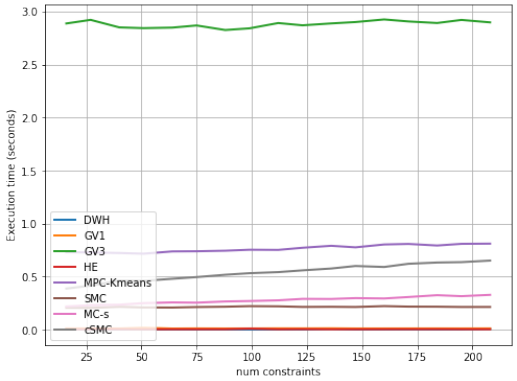
Figure 6.9: Convergence of Semi-MultiCons. Evaluation of the convergence of the Semi-MultiCons approaches in terms of the percentage of unsatisfied pairwise constraints that is represented as the cost. The horizontal axis shows the successive levels of the generated hierarchy from bottom to top, which eventually corresponds to the number of iterations, and the vertical axis shows the cost as the percentage of unsatisfied pairwise constraints.



(a) Iris (b) Wine



(c) Seeds (d) Zoo



(e) Ecoli

Figure 6.10: Comparison of execution times. The curves show the execution times of the MPC-Kmeans, DWH, HE, GV3, GV1 and Semi-MultiCons approaches for each of the Iris (a), Wine (b), Seeds (c), Zoo (d) and Ecoli (e) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the run and the vertical axis shows the number of seconds required by the approach to generate the output clustering solution.

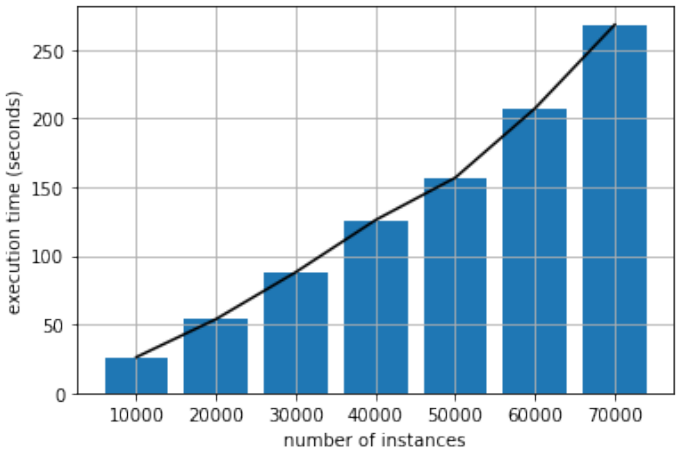


Figure 6.11: Execution times for the MNIST dataset. Each bar in the diagram represents the execution time in seconds of the cSMC Semi-MultiCons approach for the MNIST sample which number of instances is represented on the horizontal axis.

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 run. Figure 6.11 presents the average execution times for each of the seven run. The represented curve shows that the scalability property of Semi-MultiCons is linear in the number of instances for generating the output consensus hierarchy. In Section 4.3, 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 our estimate of the computational complexity of Semi-MultiCons.

6.1.11 Conclusion

The experiments conducted on five reference benchmark datasets from the UCI Machine Learning Repository and on the MNIST very large dataset have led to the following conclusions:

- The prior knowledge, i.e., supervised information, represented by the pairwise constraints 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 instances.
- Semi-MultiCons has the ability to solve real-life 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 real-world case where the appropriate 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 of clusters, as well as a consensus clustering method to achieve better quality of clustering.

Further work on Semi-MultiCons approach encompasses the integration of pairwise constraints in the closed pattern mining phase for generating constraints-based clustering patterns, and investigating the effect of noise on the performance.

6.2 Performance of Semi-MultiCons on Amadeus Datasets

As demonstrated in Table 6.1, it is infeasible to apply MPC-Kmeans on large industrial dataset as it can be extremely time consuming. We hereby only analyze the performance of the MC and SMC approach on Amadeus datasets. The experiment setting is detailed in Chapter 5. During the experiments of the Semi-MultiCons approach on the Amadeus datasets, we address the following issues:

- Performance of the Semi-MultiCons approach on large real industrial datasets.
- Comparison between the MC approach and the SMC approach to demonstrate the impact of integrating user feedback as constraints in the consensus process.
- Scalability and complexity of the Semi-MultiCons approach for real-life huge datasets.

6.2.1 Comparison between the MC and SMC Approaches on Amadeus Datasets

The experiment compares the performances, in terms of NMI and Purity indexes, and of inferred number of clusters k, for the MC and SMC approaches on Amadeus datasets. To speed up the

Table 6.2: Performance on Amadeus datasets. Comparison of the MC and SMC approaches on the real industrial Amadeus datasets of three customers for two task types. Results show the relevance of the clustering solution evaluated with the NMI (S_i) and Purity (P_i) indexes, and the inferred number of clusters k of each level i in the hierarchical consensus clustering result of Semi-MultiCons.

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.9771	**0.9786**	0.9867	**1.0000**	7100	7220
1	0.9067	**0.9072**	0.4715	**0.4739**	2182	2192
2	0.8201	**0.8204**	0.2430	**0.2437**	647	648
3	0.7355	**0.7368**	0.1452	**0.1465**	212	213
4	0.6544	**0.6622**	0.1001	**0.1034**	84	92
5	0.4558	**0.5451**	0.0545	**0.0707**	15	41

(a) Proration 1

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.9013	**0.9060**	0.9637	**1.0000**	9209	9585
1	**0.8663**	0.8653	**0.6201**	0.6190	2998	2987
2	**0.8155**	0.8152	**0.4456**	0.4446	933	917
3	**0.7429**	0.7410	**0.3399**	0.3347	300	296
4	**0.6646**	0.6644	0.2674	**0.2703**	112	114
5	0.4848	**0.5430**	0.1850	**0.2102**	19	41

(b) Proration 2

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.9686	**0.9734**	0.9428	**1.0000**	61407	65913
1	0.9125	**0.9134**	0.4426	**0.4464**	17205	17446
2	0.8472	**0.8487**	0.2216	**0.2244**	5226	5366
3	0.7883	**0.7903**	0.1278	**0.1298**	1846	1914
4	0.7317	**0.7348**	0.0815	**0.0836**	742	795
5	0.5994	**0.6438**	0.0372	**0.0493**	129	290

(c) Proration 3

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.5800	**0.5925**	0.9497	**1.0000**	18934	19713
1	0.5605	**0.5730**	0.7451	**0.7670**	4299	4378
2	0.5401	**0.5555**	0.6707	**0.6829**	1179	1205
3	0.5174	**0.5359**	0.6436	**0.6506**	437	442
4	0.4900	**0.5119**	0.6291	**0.6348**	185	189
5	0.4032	**0.5386**	0.6150	**0.6267**	37	64

(d) Interline 1

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.9081	**0.9179**	0.9255	**1.0000**	34612	36847
1	0.8459	**0.8597**	0.5063	**0.5769**	7486	7784
2	0.7738	**0.7911**	0.3337	**0.3945**	1929	2008
3	0.7161	**0.7394**	0.2619	**0.3251**	737	782
4	0.6703	**0.7021**	0.2316	**0.2947**	331	375
5	0.5693	**0.6533**	0.1939	**0.2566**	68	153

(e) Interline 2

	S_i		P_i		k_i	
Level i	MC	SMC	MC	SMC	MC	SMC
0	0.9481	**0.9518**	0.9576	**1.0000**	74366	78437
1	0.8830	**0.8850**	0.4206	**0.4301**	16236	16567
2	0.8124	**0.8155**	0.2351	**0.2412**	4336	4454
3	0.7590	**0.7638**	0.1738	**0.1792**	1680	1746
4	0.7095	**0.7202**	0.1426	**0.1487**	734	799
5	0.6004	**0.6676**	0.1089	**0.1249**	147	341

(f) Interline 3

consensus process, the enormous Amadeus datasets are divided into mini batches and the final clustering of the entire dataset is generated by appending the results of these mini batches. The number of pairwise constraints is fixed to 100 per batch and each approach is run 10 times per dataset. Results are listed in Table 6.2. Level i refers to the i^{th} consensus result in the hierarchy of Semi-MultiCons approach, from the bottom to the top, starting with 0. Columns S_i, P_i and k_i represent respectively the NMI index, the Purity index and the inferred number of clusters k of level i.

We can see that, overall, both the MC approach and the SMC approach give good performance regarding to NMI index and Purity index on Amadeus datasets. In most cases, the SMC approach outperforms the MC approach, on both NMI index and Purity index, proving the positive impact of integrating pairwise constraints into consensus process of Semi-MultiCons. For Proration 2 dataset, where the SMC approach gives slightly worse performance than the MC approach for several levels, but the SMC approach still outperform the MC approach on the bottom level, which is the recommend result to users. Especially, the SMC approach achieves remarkable purity index on the level 0, giving strong confidence to users when they determine to batch correct error ticket clusters on the bottom level.

We also find that on the bottom level, the inferred number of clusters k is larger than number of estimated classes, listed in Table 5.3, implying that the good purity index might be a trade off against inferred k. However, in real industrial scenarios, customer concerns more about the purity as they do not expect to introduce new anomalies when applying batch operation on a proposed cluster. The customer barely minds the inferred number of clusters k, except when k is extremely large, e.g. every instance is put in an individual cluster, which is not the case.

Table 6.3: Execution times for Amadeus datasets. Execution times of the MC and SMC approaches on the entire dataset, as well as on mini-batches is demonstrated. The number of instances is also given as reference.

Dataset	Number of instances	Execution Time (s)		Execution Time per Batch (s)	
		MC	SMC	MC	SMC
Proration 1	10720	**40.34**	91.01	**13.45**	30.34
Proration 2	21778	**65.98**	145.76	**13.20**	29.15
Proration 3	101524	**342.26**	701.13	**16.30**	33.39
Interline 1	39860	**63.03**	126.66	**7.88**	15.83
Interline 2	92607	**107.19**	202.66	**5.64**	10.67
Interline 3	121359	**233.44**	441.36	**9.34**	17.65

6.2.2 Scalability and Complexity Analysis

This analysis aims to evaluate the efficiency of the MC approach and the SMC approach on Amadeus datasets. Basic facts about these Amadeus datasets can be found in Table 5.3. The number of pairwise constraints was fixed to 100 per mini-batch, and 10 trials were performed for each dataset. Table 6.3 presents the average execution time for each entire dataset and the average execution time per mini-batch. We can see that execution times of both the MC approach and the SMC approach increase with the number of instances, corresponding to our analysis in Section 4.3. Compared with the MC approach, SMC has more important execution times as it integrates constraints in the consensus process. The execution time per mini-batch is less than one minute, proving the potential of the Semi-MultiCons approach to give quick or even real-time response with mini-batch mode.

6.2.3 Conclusion

The experiments conducted on Amadeus datasets from three different customers and for two task types have led to the following conclusions:

- Semi-MultiCons is able to handle enormous industrial dataset and manages to give good performance for all customers and task types.
- The SMC approach outperforms the MC approach by integrating pairwise constraints in the

consensus process to improve the quality of clustering, especially the Purity index, which is the main concern of the customer.

- The mini-batch mode improves the efficiency and the scalability of Semi-MultiCons and makes it possible for Semi-MultiCons to give quick or even real time response.

Future work on Semi-MultiCons application to Amadeus datasets includes collecting real must-link and cannot-link constraints from end-users to obtain concrete result of the SMC approach, and extending the experiments to more customers and task types to fully investigate the performance of Semi-MultiCons on Amadeus data.

Chapter 7

Proposed Task Handling Module based on Semi-MultiCons

The objective of this thesis is to contribute to the improvement and automation of the error correction process of Amadeus Revenue Accounting System, as stated in Chapter 1.2. In the previous chapter, we presented the proposed Semi-MultiCons approach and demonstrated its performance on benchmark UCI MLR datasets compared to other state-of-the-art methods. In this chapter, we design a Proof-of-Concept prototype which applies Semi-MultiCons approach on Amadeus Revenue Accounting Workflow to demonstrate the practical potential of the proposed solution as a Task Handling Module tool in real industrial scenarios.

7.1 Task Correction Process with Current Task Handling Module

In Chapter 1.1, we briefly introduced the notation of Amadeus Revenue Accounting Workflow (RAW), and we mainly focused on explaining why and how tasks are raised. In this section, we describe in detail the mechanism of current Amadeus Task Handling Module to illustrate how tasks are solved.

Figure 7.1 shows the task correction mechanism with current Task Handling Module. each arrow shows a data flow from one entity to another in the direction indicated by the arrow. The meaning of icons is labeled in the figure. Generally, data with anomaly pattern is marked as purple while normal data is blue. Different data flows are listed with number in circle. We hereafter explain the task correction process illustrated in Figure 7.1:

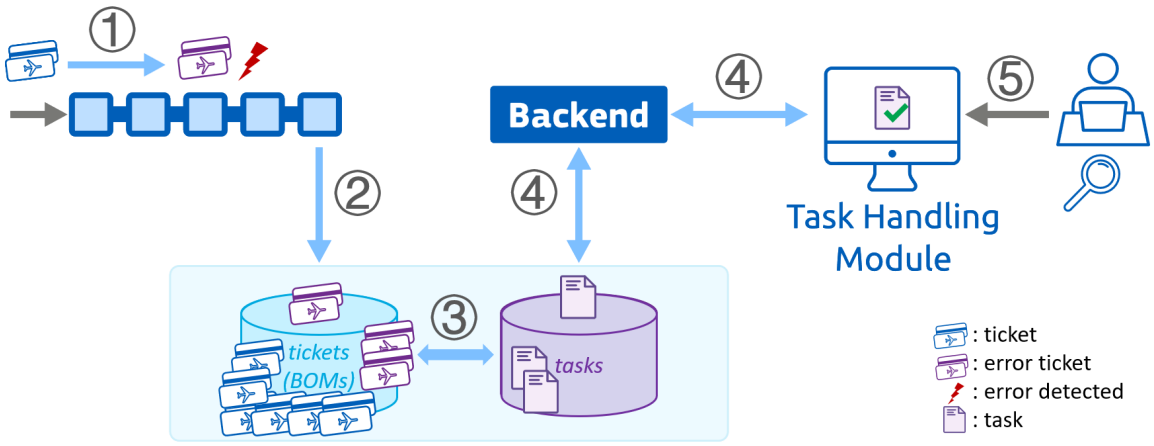


Figure 7.1: Task correction mechanism with current Task Handling Module. Revenue Accounting System automatically processes tickets until an error occurs. The workflow is then interrupted, and a task associated with the error ticket is raised. The user is then required to manually investigate and correct the tasks one by one through the Task Handling Module.

- Data flow 1, 2 and 3 simplify the revenue accounting workflow shown in Figure 1.1. RAW automatically processes ticket data until an error is detected. The workflow is then interrupted and a task associated with the error ticket is raised. Both tickets and tasks are represented as BOMs in the database. However, tasks do have some additional information such as task creation time, task type (from which module the task is raised), etc.

- Data flow 4 demonstrates the capacity of the current Task Handling Module (THM) to interact with task database, so that THM is able to display relevant data to the user or apply user correction on error ticket's BOM.

- Data flow 5 represents the action from user. Each time, the user can only access one task. Even if similar tasks have already been corrected, the user is required to do repeated work.

The main issue of the current Task Handling Module comes from data flow 4 and 5, where each task is treated as independent. Therefore manual effort and time are wasted on tasks that have similar corrections. To address this issue, we apply the developed Semi-MultiCons approach on THM and propose a novel solution.

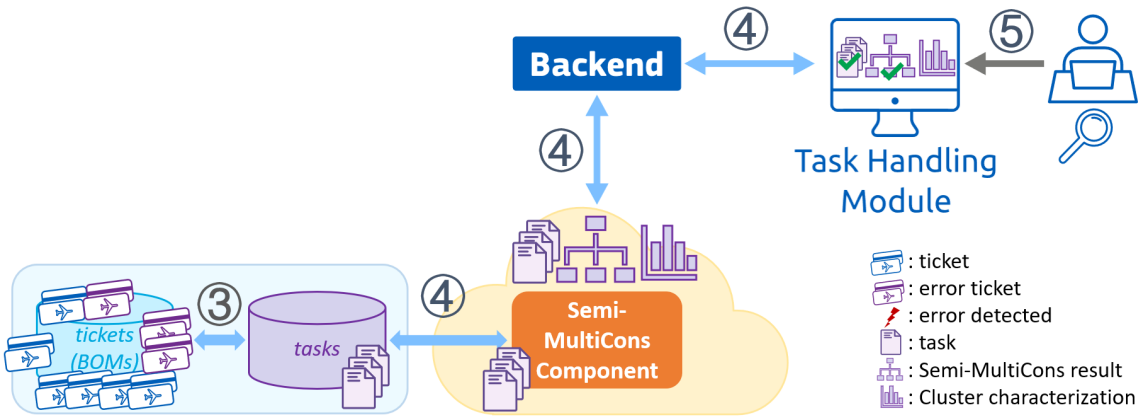


Figure 7.2: Proposed Task Handling Module solution based on Semi-MultiCons. The user is able to explore task clusters that contain tasks that are similar with the current one. Cluster characterization is provided as well, to assist the user in the validation of clusters and make batch corrections.

7.2 Proposed Solution Based on Semi-MultiCons

We propose a new Task Handling Module based on Semi-MultiCons, displayed in Figure 7.2. Novel functionalities, which are marked as orange, are presented in the following:

- Data flow 1, 2 and 3 remain the same as current THM and are not fully demonstrated due to page limitation.
- A component implementing Semi-MultiCons approach is added. It gives user additional access to Semi-MultiCons hierarchical clustering result and characterization of clusters, as illustrated by data flow 4. To guarantee performance and efficiency regards to the huge size of industrial datasets, the component is deployed on Cloud.
- Initially, user is still only allowed to access one task in data flow 5. However, it is possible to explore similar tasks from the current one. User is expected to make batch corrections on cluster of tasks, based on Semi-MultiCons hierarchical clustering result and characterization of clusters provided by the component. See Section 7.4 for concrete examples.

Figure 7.3 shows in detail the interactive process between the user and Semi-MultiCons component. On the one hand, it requires from the component to generate hierarchical results and to provide cluster characterization based on feature importance computed by Random Forest. On the other hand, it also requires from the end-user to configure the data pre-processing step and the base

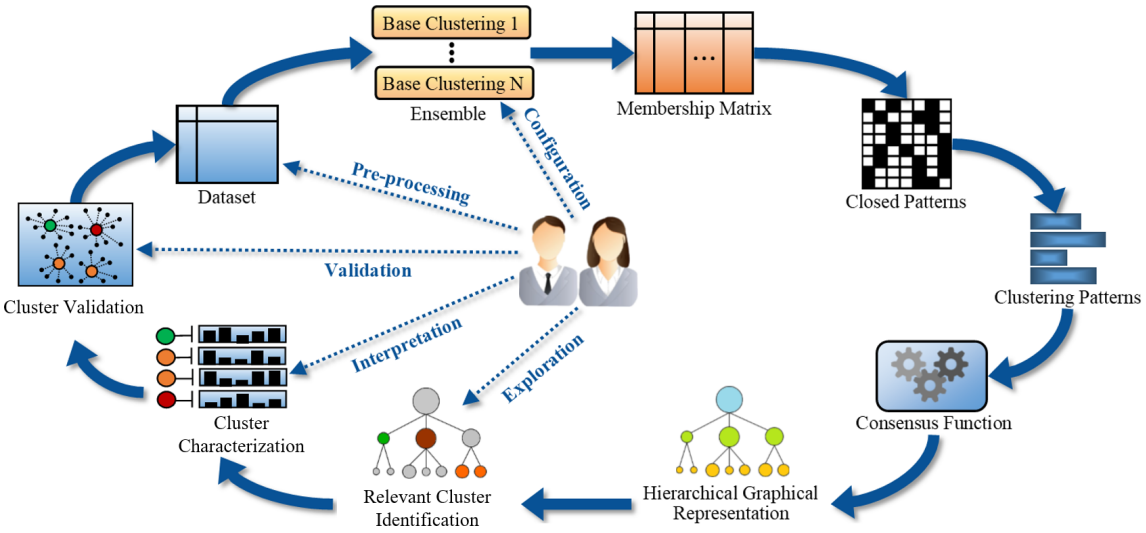


Figure 7.3: Interactive process between user and Semi-MultiCons component. The component generates hierarchical clustering results and cluster characterizations for the user to explore, interpret and validate. Meanwhile, the user is required to define the algorithmic configurations of base clusterings for generating the binary membership matrix in input of the Semi-MultiCons approach.

clustering algorithmic configurations, and to explore, interpret and validate the task clusters, i.e., identify and select the most relevant clusters from Semi-MultiCons hierarchical results. Validated task clusters can then be integrated in the process as supervised information in the form of pairwise constraints.

7.3 Design of Semi-MultiCons Component

Current Task Handling Module is implemented in C++ with HTML and CSS. The BOM data about tickets and tasks is available in Oracle database on-premises. However, to achieve better efficiency and scalability, the Semi-MultiCons approach requires the usage of Big Data ecosystem and Cloud platform. The designed component must be able to access the database on-premises, meanwhile, it must also implement the interactive operations listed in Figure 7.3. The component has three parts:

- The *data collection part* is responsible for preparing raw data for the component. It is scheduled to run daily, acquiring incoming tickets and tasks data during the day from database on-premises. This part ensures the access to the database on-premises.

- The *core part* is the implementation of Semi-MultiCons. Raw data is pre-processed and hierarchical clustering result of tasks is generated. User is able to interact with Core part, i.e. configure input parameters and get clustering results, via RESTful API.
- The *cluster characterization part* characterizes selected clusters through the analysis of their discriminating features. Using clusters as class labels, a Random Forest is constructed to identify the features that distinguish each cluster from the others in the data space. Like core part, user interactive operations are available through RESTful API.

The component is deployed on OpenShift Container Platform of Amadeus and is implemented in Spark with R language. RESTful API is programmed by using Python with Kafka to communicate messages between on-premises and Cloud.

7.4 Industrial Scenarios

To illustrate the component we designed, we create a Proof-of-Concept prototype with Graphical User Interface (GUI) using Angular. Different real-life scenarios are presented in this section to demonstrate the practical potential of the proposed solution as a Task Handling Module tool for Amadeus Revenue Accounting System, to improve the automation of the error correction process. Sample tickets and tasks in scenarios are retrieved from Amadeus database, and sensitive data has been anonymized.

7.4.1 Access One Task and Make Correction

Figure 7.4 presents the GUI of our prototype. Tasks are summarized in a table and user can access one of them by clicking on it. In task detail page shown in Figure 7.5, relevant information regards to the task is displayed. Investigating these information helps user to understand the task and a correction can be made by changing editable values. The correction will be dispatched to ticket and task database once the 'Solve task' button is clicked.

7.4.2 Explore Similar Tasks from the Current One

Besides single task correction, our proposed solution allows user to explore similar tasks from the current one, as illustrated in Figure 7.6. A similar tasks table is available at the bottom of task detail page. By default, only the lowest level of Semi-MultiCons hierarchical clustering result is displayed.

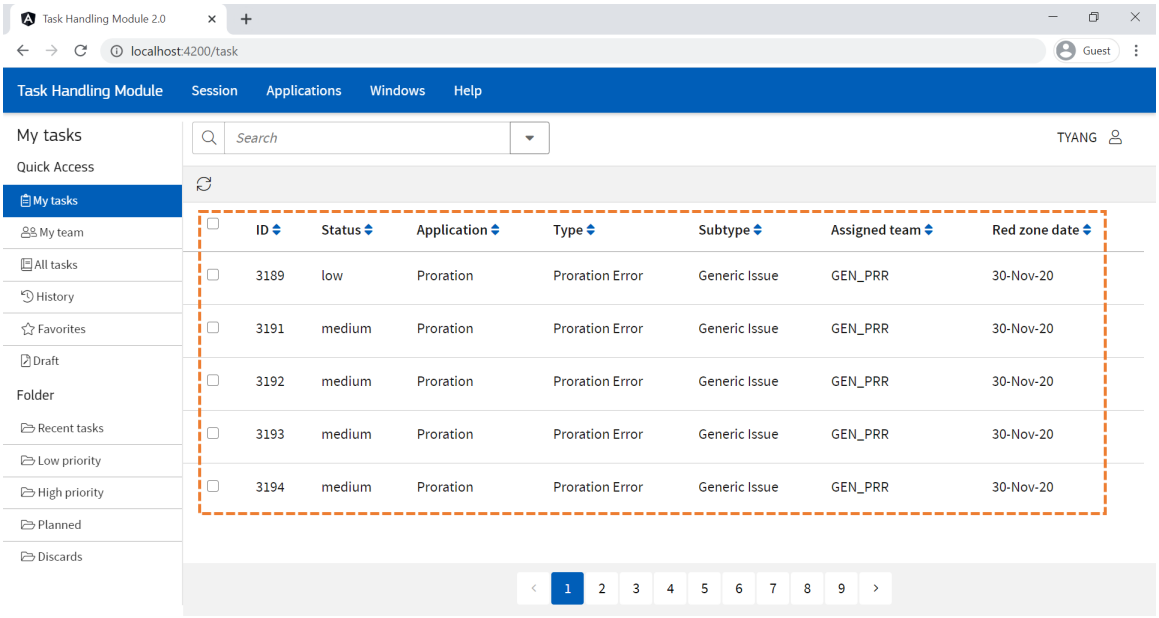


Figure 7.4: Screenshot of task list. Tasks are listed in table to give an overview of each task. Clicking on a task will redirect the user to the task detail page.

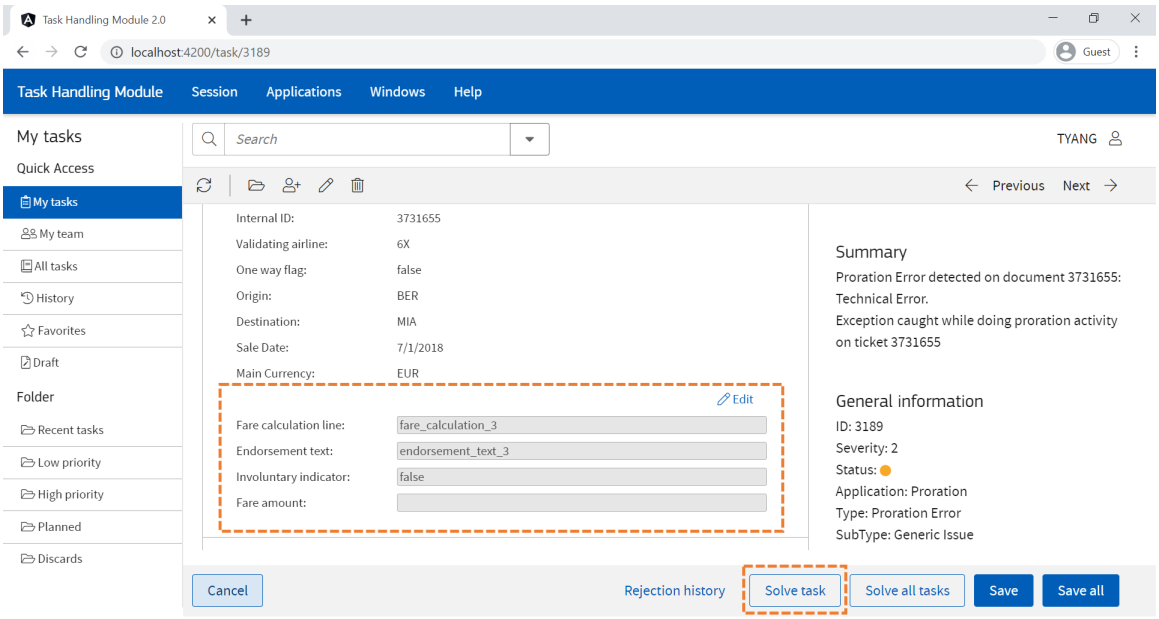


Figure 7.5: Screenshot of task details. User can correct editable values in task. The correction is dispatched to database once the 'Solve task' button is clicked.

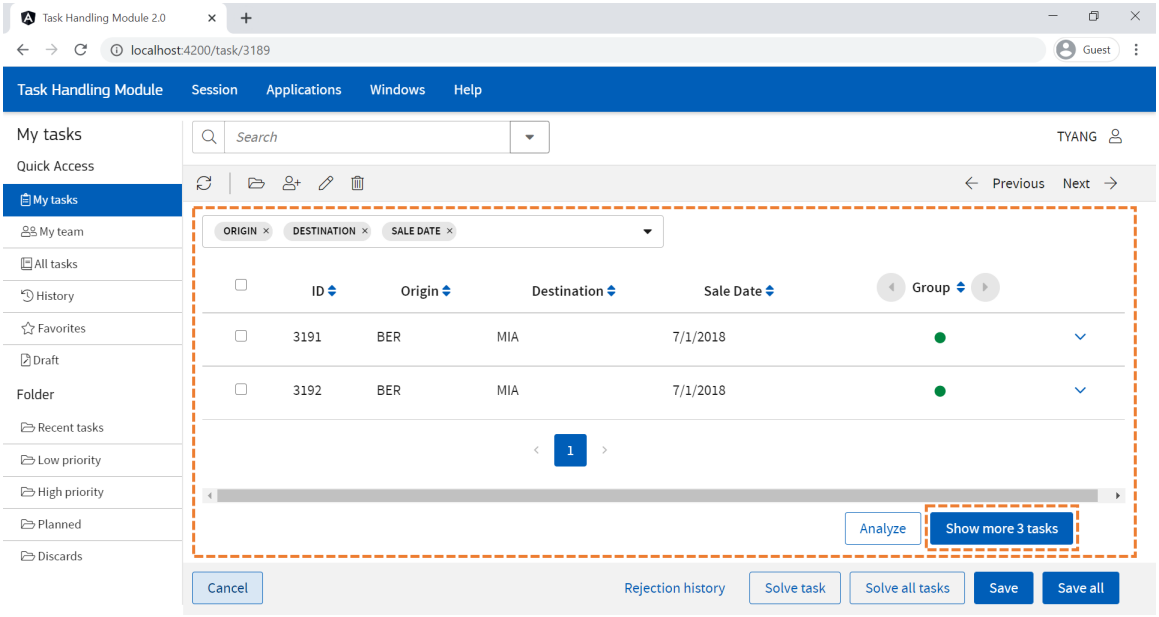


Figure 7.6: Screenshot of similar task list. Tasks that are similar with the current one are summarized in the similar task table at the bottom of the task detail page. The similar task table can be expanded by clicking on the 'Show more' button.

Similar task table can be expanded by clicking on the 'Show more' button as demonstrated in Figure 7.7. Tasks which are merged with the current cluster in upper level are added to the table. More precisely, assume that the current task is x_4 in Figure 4.3. Only x_5 is displayed in similar task table by default. Clicking once on 'Show more' button ends up to adding x_6, x_7 to the table.

Color in Group column of similar task table represents cluster ID and overall cluster similarity with current task. Concretely, if current task is x_4 in Figure 4.3. x_5 is displayed by default and cluster $\{x_4, x_5\}$ is very similar to x_4, thus marked as green. After clicking on Show more button, x_6 and x_7 are added. They belong to cluster $\{x_6, x_7\}$ in lowest level result and the overall similarity is lower than $\{x_4, x_5\}$, therefore marked as yellow. We can switch to upper or lower level clustering result through clicking right arrow or left arrow button next to Group. Clicking right arrow button on Figure 7.7 results in Figure 7.8. Back to our previous example, as cluster $\{x_4, x_5\}$ and cluster $\{x_6, x_7\}$ are merged in upper level, x_5, x_6 and x_7 will change their color to orange if right arrow button is clicked. The 'Show more', left arrow and right arrow buttons allow the user to navigate and explore similar tasks in the Semi-MultiCons hierarchical clustering result.

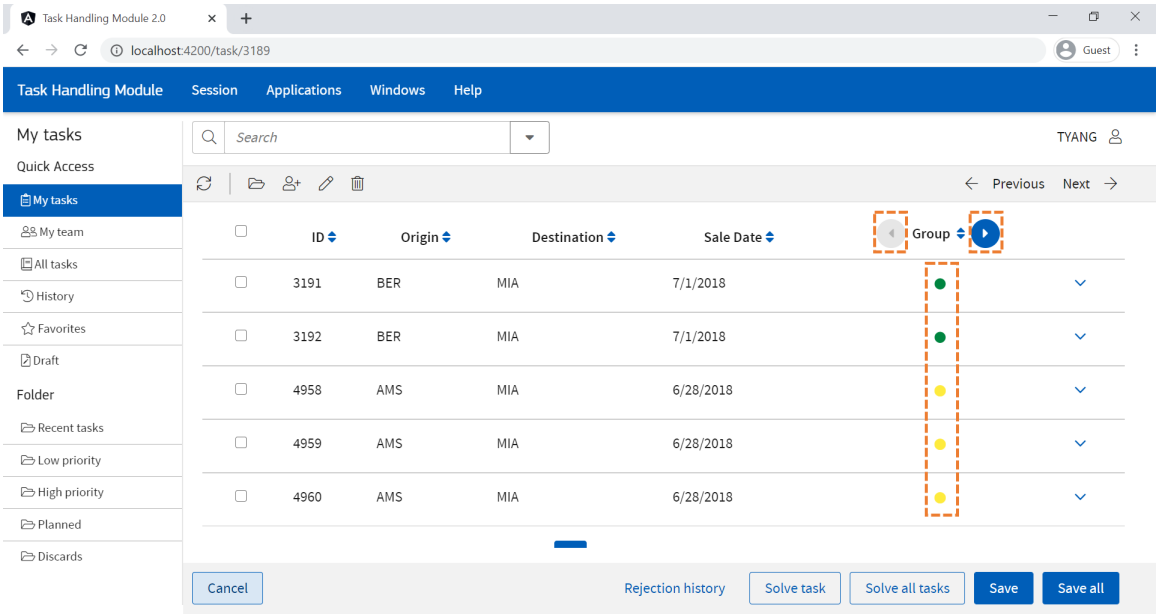


Figure 7.7: Screenshot of group column in similar task list. The color in the Group column of the similar task table represents cluster ID and the overall cluster similarity with the current task.

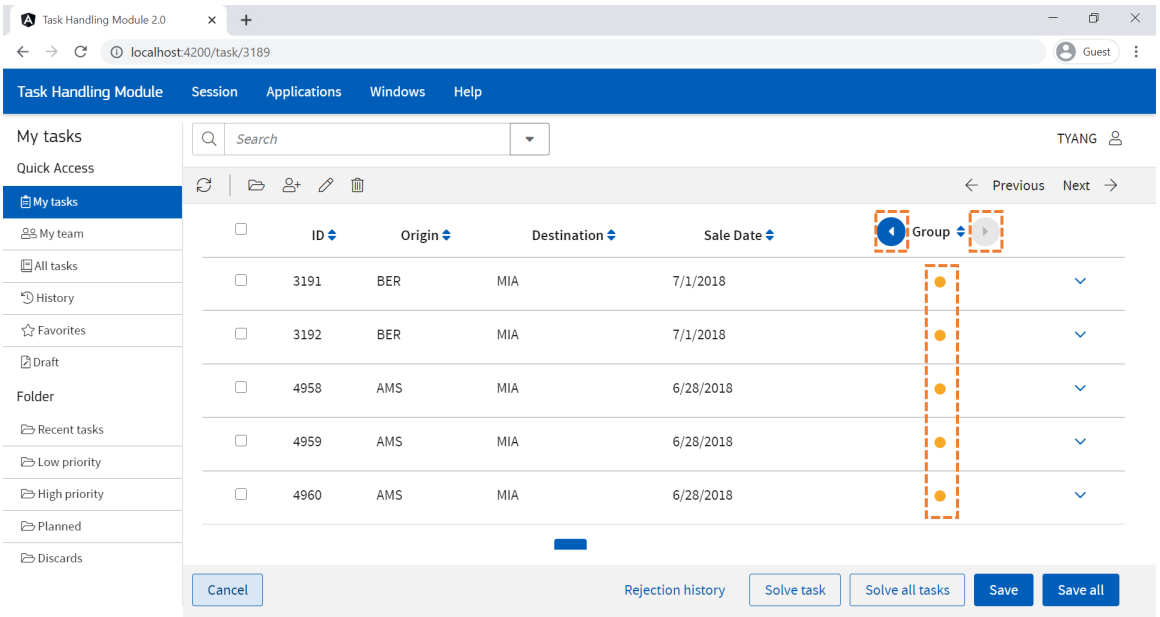


Figure 7.8: Screenshot of arrow buttons next to Group column. User can switch to upper or lower clustering results, that is sub-level or sup-level in the Semi-MultiCons hierarchical result, through clicking right arrow or left arrow button next to the Group column.

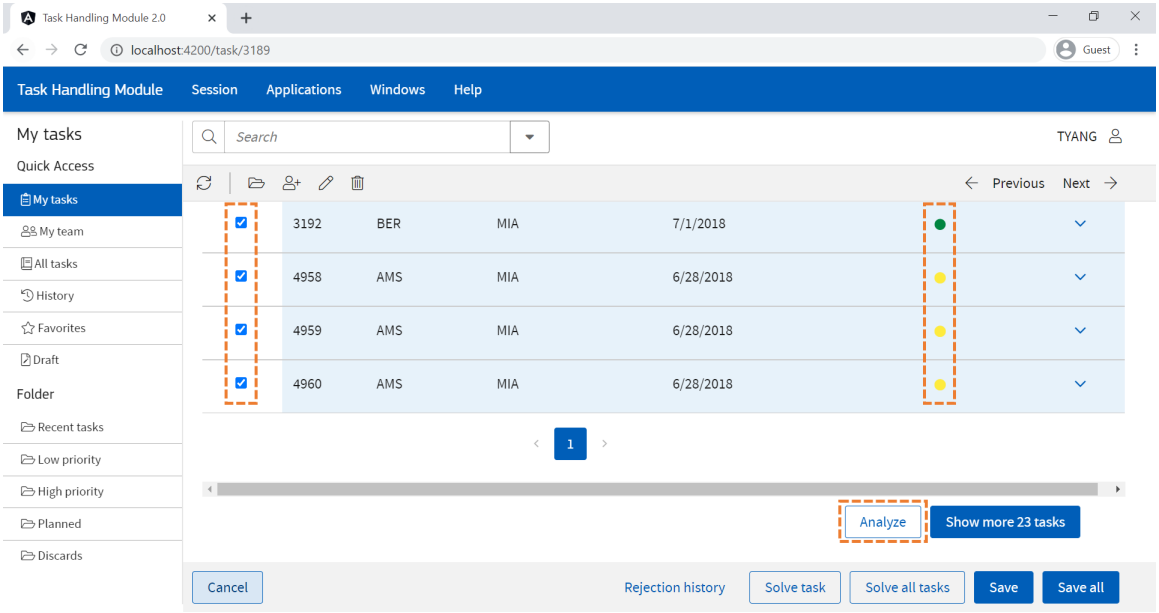


Figure 7.9: Screenshot of the 'Analyze button' result. User can select one or several tasks in the similar task table and click on the 'Analyze button' to request for the analysis of cluster features based on current clustering result.

7.4.3 Cluster Characterization

Characterization of clusters is an indispensable step to assist user in the understanding of the underlying reasons why tasks were grouped or separated in clustering results. User can select one or several tasks in the similar task table, as presented in Figure 7.9, and click on the 'Analyze button' to request for the features that distinguish each cluster from the others in the data space. An example result is shown in Figure 7.10. Moreover, additional features can be displayed in the similar tasks table through the drop down selector on top of the table (see Figure 7.11) to give the user the possibility to investigate and verify the features provided by the 'Analyze button'.

7.4.4 Batch Correction

The user is able to make batch correction on similar task group, as shown in Figure 7.12. One or several tasks in the similar task table can be selected and once the 'Solved all tasks' button is clicked, the correction for the current task will be dispatched to all selected tasks.

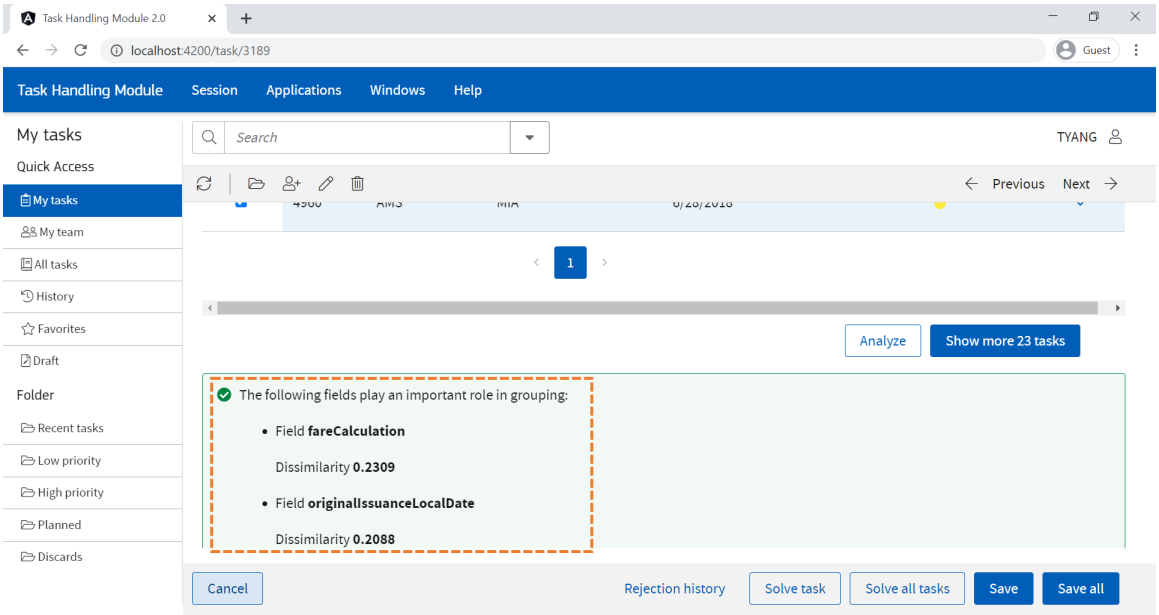


Figure 7.10: Screenshot of example feature analysis result. The name of the attributes which play an important role during the clustering process are provided, as well as their importance factor.

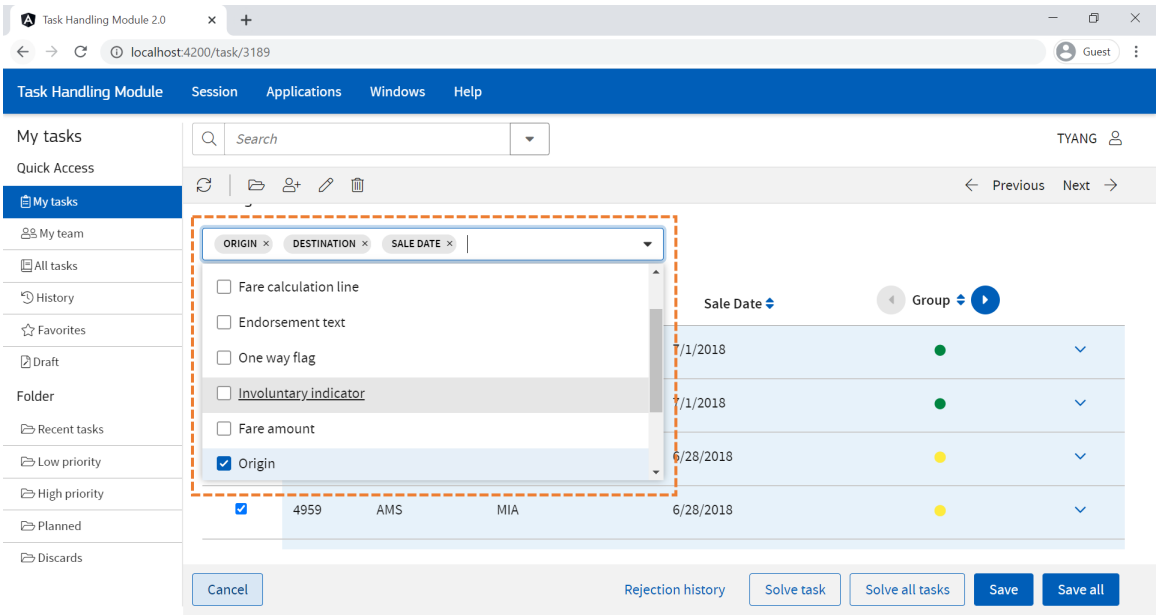


Figure 7.11: Screenshot of displayed feature selector. Additional features can be displayed in the similar task table through the drop down selector on top of the table.

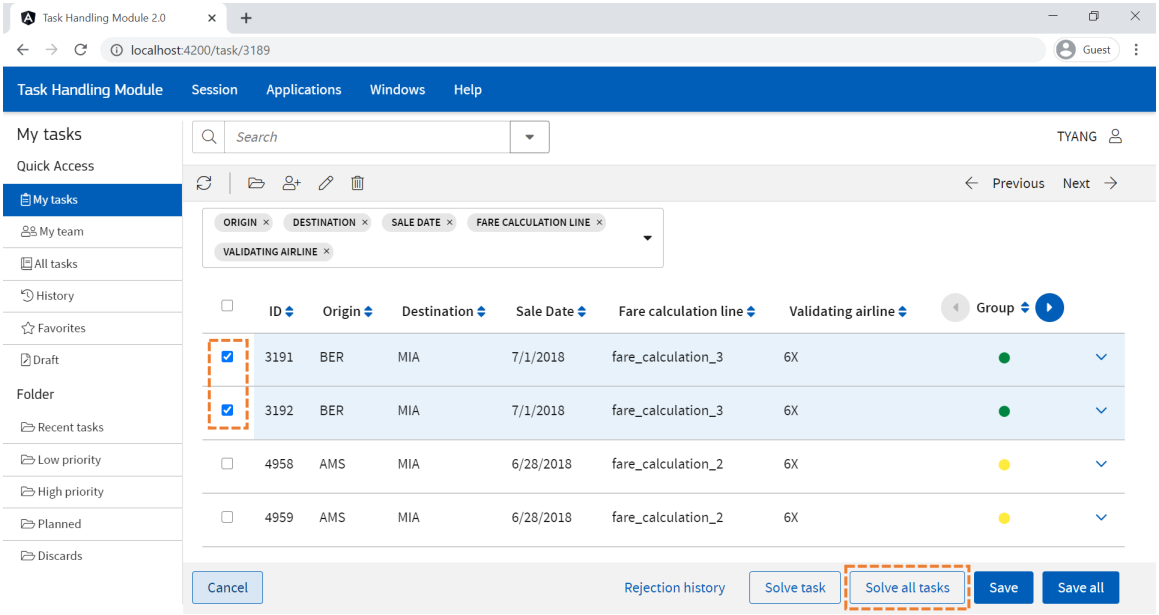


Figure 7.12: Screenshot of the 'Solve all tasks' button. One or several tasks in the similar task table can be selected and once the 'Solved all tasks' button is clicked, the correction performed on the current task will be dispatched to all selected tasks.

7.5 Conclusion and Future Work

In this chapter, we propose a novel Task Handling Module tool for Amadeus Revenue Accounting System based on the Semi-MultiCons approach. A new Semi-MultiCons component is integrated to the current THM to improve the automation of the error correction process and to emphasize user interactive operations during the process. We create a Proof-of-Concept prototype of the proposed solution with Big Data ecosystem and Cloud platform. With this novel THM tool, the user is able to explore clusters of similar tasks in the Semi-MultiCons hierarchical clustering result, to request for the analysis of features explaining why tasks were grouped or separated, to validate similar task clusters, as well as to make batch fixes or corrections per cluster. The validation data from the user will then be used as supervised information to improve the overall quality of Semi-MultiCons clustering results later on.

The final objective of this application is to be integrated as a novel Task Handling Module of Amadeus Revenue Accounting System. The prototype presented here is currently under test phase. As future work, we plan to evaluate and adapt our solution according to feedback from Revenue Accounting domain experts and Amadeus customers. The evaluation will focus in particular on the

accuracy of similar task clusters as well as on the time spent for performing corrections using the proposed solution compared to the time spent using the current Task Handling Module.

Chapter 8

Impact of Unbalanced and Noisy Constraint Set

8.1 Introduction

Clustering methods aim at grouping data into homogeneous groups, such that instances in the same group are more similar to each other than to those in other groups [85, 86]. As an unsupervised learning approach, clustering is often used to analyze datasets that lack any supervisory information [76]. However, in practice, we may have some prior knowledge available about the underlying clustering, for instance, a few numbers of labels or a set of constraints. In order to take advantage of this supervisory information and extract more relevant information for the user, recent research works focused on semi-supervised clustering, also called constrained clustering, which uses prior knowledge to guide the clustering process [25, 31].

Different types of prior knowledge and semi-supervised clustering approaches are considered in the literature. The most widely used type of constraints is the instance-level must-link/cannot-link constraint [75]. A must-link constraint implies that two instances should be assigned to the same group. On the other hand, a cannot-link constraint enforces that two instances cannot be placed in the same group. Many semi-supervised clustering approaches have been developed based on must-link and cannot-link constraints. Most of them extend classical clustering algorithms, such as the K-means algorithm, to a constrained version [8, 17, 32, 56, 74, 75]. Some other methods focus on using must-link and cannot-link constraints to infer new similarity metrics [5, 22, 58, 84] or to model the problem by using a declarative framework [20, 33, 35, 54]. Other methods integrate

constraints into a collaborative [30] or ensemble-based process [3, 42, 82] that involves several clustering algorithms. Recently, some studies propose to make use of constraints in deep clustering methods [59, 102].

While most work in the literature focus on the development of novel and efficient semi-supervised clustering variants, the impact of noisy input on semi-supervised clustering algorithms has not been explored in-depth. Most research work assume that the input of the algorithm, that is for instance the number of clusters k and the constraint set, is accurate. The input constraint set is usually generated based on ground truth labels of instances and is "balanced". A more realistic situation in an industrial context, where noisy constraints exist and the distribution of constraints can be extremely biased, is barely considered to the best of our knowledge. In [17], the authors analyze the complexity of their proposed semi-supervised clustering approach in the case where the constraint set is skewed. In [21], the impact of constraint set characteristics on performance is pointed out, proving that some constraint sets can actually decrease algorithm accuracy. This leads to further work that concentrate on how to select informative and/or "easy" constraint sets [18]. Among the few articles in the literature which discuss the impact of noisy constraints and of an incorrect number of clusters k, we can cite [13, 53, 101] for the first issue, and [101] for the second one. In [4] and [14], the behaviour of semi-supervised clustering algorithms in presence of erroneous constraints is studied.

In this chapter, we simulate realistic industrial settings where the input is noisy, which means the constraint sets are skewed or contain noisy constraints. We analyze the robustness and accuracy of the Semi-MultiCons approach and of each other semi-supervised clustering approach, and highlight the scenarios for which each approach is more suitable.

8.2 Semi-supervised Clustering Approaches

In this section, we present the six semi-supervised clustering algorithms used in these experiments. These include three K-means algorithm variants, namely COP-Kmeans [75], PC-Kmeans and MPC-Kmeans [8], that integrate constraints in classical K-means algorithm, and three metric learning approaches, namely RCA [63], MMC [84] and ITML [22], that use constraints to learn a distance function.

8.2.1 Constrained K-means

Constrained K-means clustering (COP-Kmeans) was developed in [75] and is one of the most prominent constrained clustering algorithms. The idea is to ensure that none of the constraints is violated. An instance i is directly assigned to a cluster if the cluster contains an instance which has a must-link constraint with i. Otherwise, i will be assigned to the cluster that has the closest centroid, excluding the clusters containing an instance that has a cannot-link constraint with i. If a legal cluster cannot be found, then an empty partition is returned.

COP-Kmeans is recognised as being both simple and efficient. However, since COP-Kmeans enforces that each constraint is satisfied, noisy constraints will inevitably lead to noisy output. COP-Kmeans can also be sensitive to instances assignment order. Once an instance is assigned to a cluster, all the other instances that have must-link with it will be assigned to the same cluster. A different assignment order may thus end up in a different clustering results for those instances. COP-Kmeans may return an empty partition when an instance cannot be assigned to any cluster due to having a cannot-link constraint with instances among all clusters.

8.2.2 Pairwise Constrained K-means and Metric Pairwise Constrained K-means

Based on the idea of using constraints in K-means algorithm, [8] proposed Pairwise Constrained K-means (PC-Kmeans) and Metric Pairwise Constrained K-means (MPC-Kmeans). PC-Kmeans utilizes constraints for seeding the initial clusters and it directs the instance assignments to clusters to respect the constraints. The connected components, which consists of instances connected by must-link constraints, are taken as initial cluster centroids. The objective function is formulated as the sum of the total squared distances between instances and their cluster centroids, and the penalty induced by violating any constraint. During the cluster assignment step, an instance will be assigned to the cluster centroid which minimizes the objective function.

MPC-Kmeans involves both cluster initialization, cluster assignment and metric learning in a unified framework. The objective function definition and cluster initialization are the same as for PC-Kmeans. Still, the distance metric is adapted by re-estimating the weight matrices during each iteration based on the current cluster assignments and constraint violations.

PC-Kmeans and MPC-Kmeans never return an empty partition as COP-Kmeans sometimes do. But as variants of Kmeans algorithm, they can be order sensitive as well. The complexity of MPC-Kmeans is critical [14] since MPC-Kmeans updates weight matrices during each iteration of the clustering process.

8.2.3 Relevant Components Analysis

Relevant Components Analysis (RCA) [63] is one of the earliest methods that integrates constraints in metric learning. It makes use of "chunklet", which is is essentially equivalent to connected component introduced in Chapter 3, to compute a global linear transformation to assign large weights to relevant dimensions and low weights to irrelevant dimensions [89]. This transformation is based on chunklet information only, and does not use any cannot-link constraints.

8.2.4 Mahalanobis Metric for Clustering

Mahalanobis Metric for Clustering (MMC) [84] aims to minimize the sum of Mahalanobis distances between instances linked by must-link constraints, and at the same time, enforce the distances between instances linked by cannot-link constraints to be greater than a constant (often set to 1). This distance metric is trained using convex optimization, and the training process is thus local optima free.

Although the MMC approach is efficient, the computation of eigenvalues during metric learning step can be sometimes time consuming. Another restriction is its unrealistic assumption that all clusters follow a unimodal distribution. Also, MMC is reported to have some uncertainty about the optimality of the final solution. The proposed gradient based algorithm of MMC needs tuning several parameters, and it is not guaranteed to find the optimum without such tuning [5].

8.2.5 Information-Theoretic Metric Learning

Similarly to MMC, the Information-theoretic metric learning (ITML) approach [22] aims to learn an optimal Mahalanobis distance subject to constraints. It gives a bijection between the Mahalanobis distance and an equal-mean multivariate Gaussian distribution. In this way, the problem is translated to minimizing the differential relative entropy, also known as Kullback-Leibler divergence, between two multivariate Gaussians under constraints on the distance function. The problem is then expressed as a particular Bregman optimization problem by minimizing the LogDet divergence subject to linear constraints.

Unlike some other metric learning methods, no eigenvalue computation or semi-definite programming is required in ITML. It can also handle a wide variety of constraints, and can optionally incorporate a prior on the distance function. However, a simple bijection between Mahalanobis distance and equal-mean multivariate Gaussian distribution oversimplifies the underlying metric struc-

ture. In practice, there will often be no feasible solution to the general ITML problem, particularly when the number of constraints is large, as reported in [47].

8.3 Experimental Setting

The Iris, Wine and Seeds UCI MLR benchmark datasets, presented in Table 5.1, were used in these experiments. The Zoo and Ecoli datasets were excluded for the reason that the distribution of their classes is unbalanced. As demonstrated and explained in Section 6.1.2, the existence of minority class may result in a mismatch between number of classes and the real number of clusters in dataset, leading to a confusion about the choice of the number of clusters k parameter value for semi-supervised clustering approaches.

The six semi-supervised clustering approaches presented in the previous section, and the SMC approach were selected for these experiments. Implementations of the semi-supervised clustering approaches can be found in Python packages active-semi-supervised-clustering [66] and metric-learn [23]. These approaches require in input the number of clusters k, for which the number of classes is given.

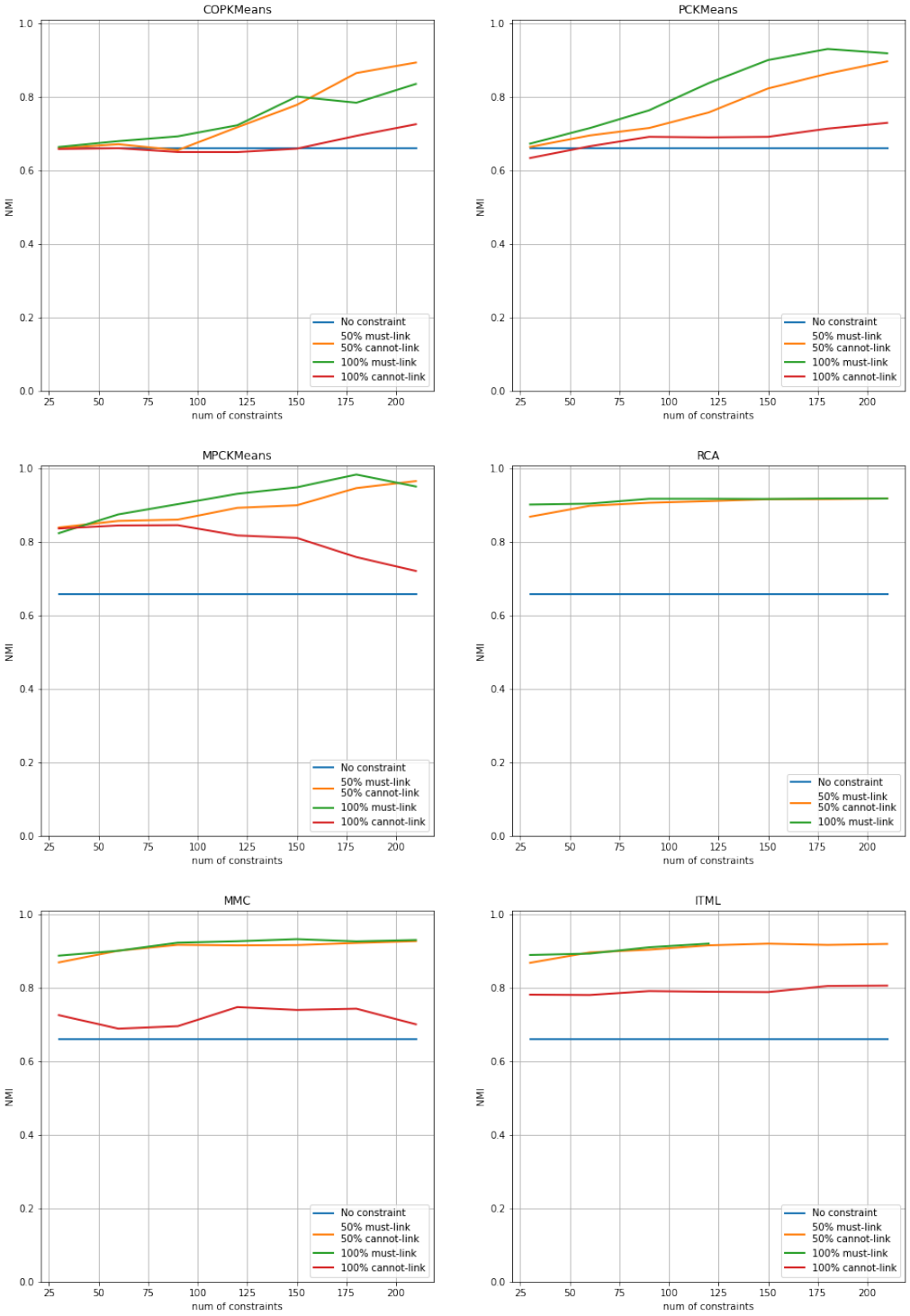
The SMC approach with Kmeans base clustering was selected to demonstrate the impact of unbalanced and noisy input on the consensus process of Semi-MultiCons. The MC-s approach and the cSMC approach were not included in these experiments, since constraints are integrated in both the ensemble creation and the consensus generation steps of these approaches, making it difficult to distinguish the specific influence of noisy input on each step. The input parameters of the SMC approach are detailed in Table 5.2. The semi-supervised clustering approaches and the SMC approach were evaluated using the NMI index.

8.4 Experimental Results

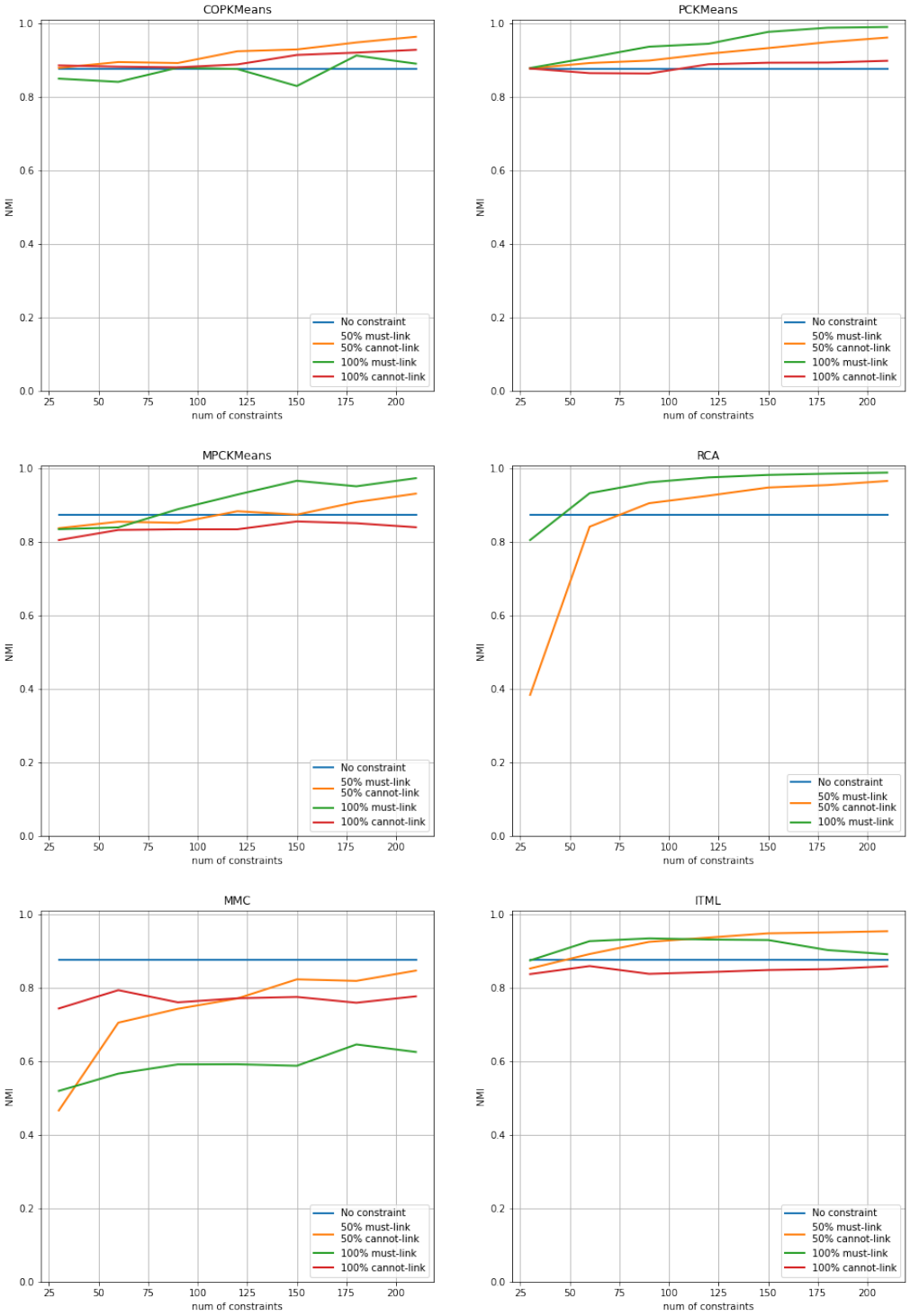
The experimental results about the impact of noisy input on the performance of semi-supervised clustering approaches and Semi-MultiCons are reported in this section.

8.4.1 Impact of Unbalanced Constraint Sets on Semi-supervised Clustering Approaches

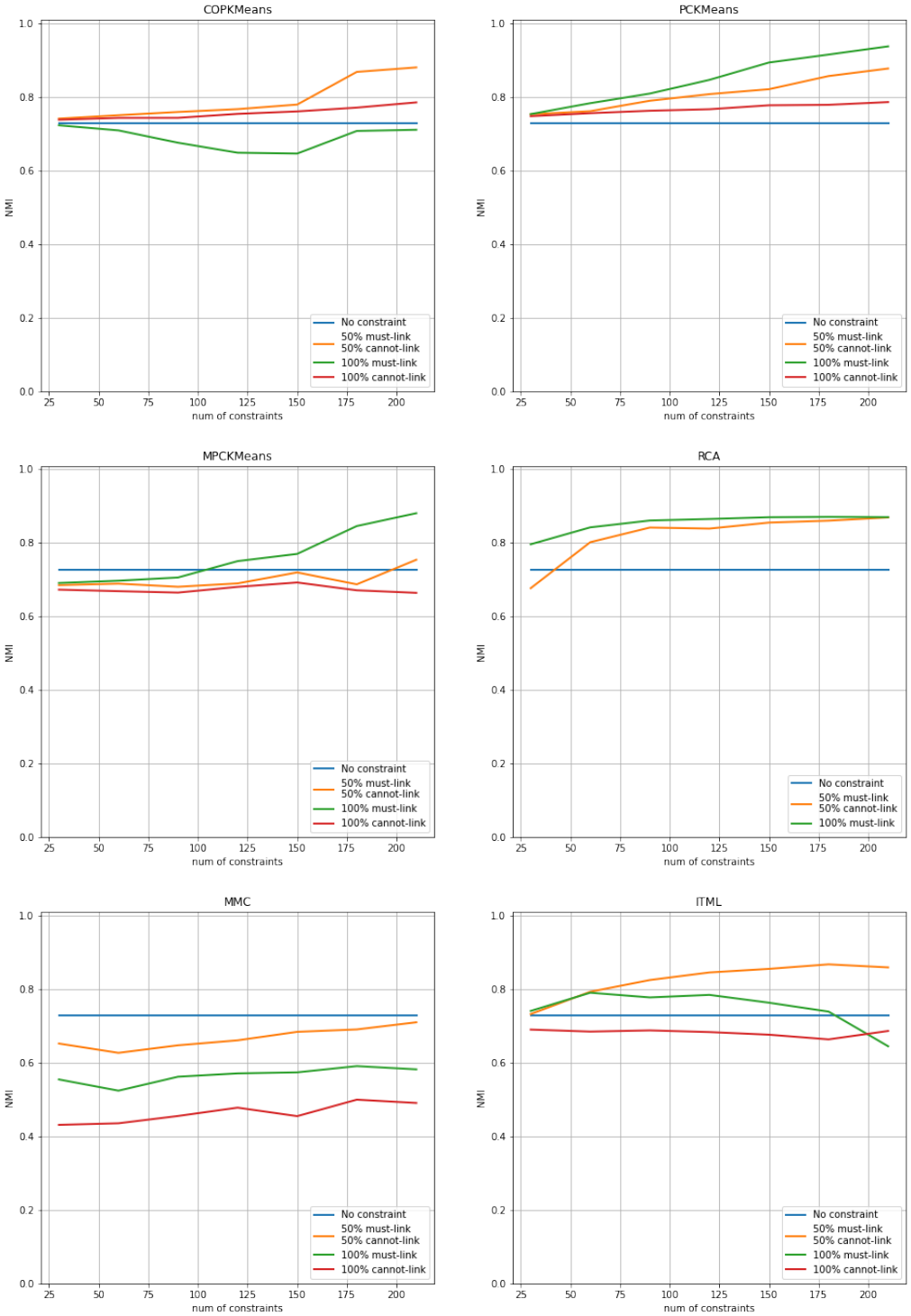
During this experiment, we analyse the impact of highly unbalanced constraint sets on the semi-supervised clustering approaches. The number of constraints ranges from 0 to 210. For each number of constraints, 30 different constraint sets were generated to obtain repeated trials. The


COPKMeans
PCKMeans
MPCKMeans
RCA
MMC
ITML
NMI
num of constraints
No constraint
50% must-link
50% cannot-link
100% must-link
100% cannot-link

(a) Iris


COPKMeans
PCKMeans
MPCKMeans
RCA
MMC
ITML
NMI
num of constraints
No constraint
50% must-link
50% cannot-link
100% must-link
100% cannot-link

(b) Wine



(c) Seeds

Figure 8.1: Performance of semi-supervised clustering with unbalanced constraint sets.

generated constraint sets contain either 50% must-link constraints and 50% cannot-link constraints, only must-link constraints, or only cannot-link constraints, to separately investigate their impact on performances.

The results of COP-Kmeans, PC-Kmeans, MPC-Kmeans, RCA, MMC and ITML for the Iris (a), Wine (b) and Seeds (c) datasets are presented in Figure 8.1. The horizontal axis shows the total number of pairwise constraints used during the run and the vertical axis shows the average NMI index score of the output clustering solution over all trials for each approach. The vertical axis is normalized to a [0.0, 1.0] range for all subplots in the figure. The blue curve corresponds to the unsupervised clustering result when no constraint is used, with input parameter k set to number of classes. The orange curve corresponds to the NMI index score of each approach in the case the number of must-link constraints is equal to the number of cannot-link constraints. The green curve and red curve represent the NMI evaluation of each approach in the situation where the constraint set contains respectively only must-link constraints and only cannot-link constraints.

As demonstrated by the red curve for COP-Kmeans, it may return an empty partition sometimes, when an instance cannot be assigned to any cluster. The PCKmeans approach does not have good performance on the Wine dataset, which is similar to results presented in the original PCKMeans paper [8]. The red curve for the RCA approach is not illustrated because this approach does not use cannot-link constraints. For the Wine and Seeds datasets, the MMC approach does not find the optimal final solution, proving the statement in [5] that the MMC approach needs tuning of several parameters and is not guaranteed to find the optimum without such tuning. The ITML approach fails to find a feasible solution for the Iris dataset when the number of constraints is large, as represented by the green curve. This corresponds to the report in [47], arguing that there will often be no feasible solution to the general ITML problem in practice, particularly when the number of constraints is large.

Comparing the blue curve with the orange, green and red curves, we can clearly see that the negative effect, explained in Section 6.1.7, widely exist in semi-supervised clustering approaches, especially when number of constraints is small. Overall, the semi-supervised clustering approaches benefit more from must-link constraints than from cannot-link constraints. A pure cannot-link constraint set, as demonstrated by the red curve, usually leads to a decrease in performance, especially for COP-Kmeans and MPC-Kmeans. In contrast, must-link constraints has a significant positive impact on the performance. Most approaches achieve their best performance with only must-link constraint set, as illustrated by the green curve. The ITML approach and the MMC approach probably have the ability to make use of cannot-link constraints, since their orange curve has better

performance than the green curve.

The metric learning methods RCA, MMC and ITML converge faster than K-means variants COP-Kmeans, PC-Kmeans and MPC-Kmeans, as their curves quickly reaches their peak of performance when the size of constraint set is small. The performance then remains stable, even when the number of constraints increases. However, K-means variants, especially MPC-Kmeans, seems to have a higher NMI index score when the number of constraints is large enough.

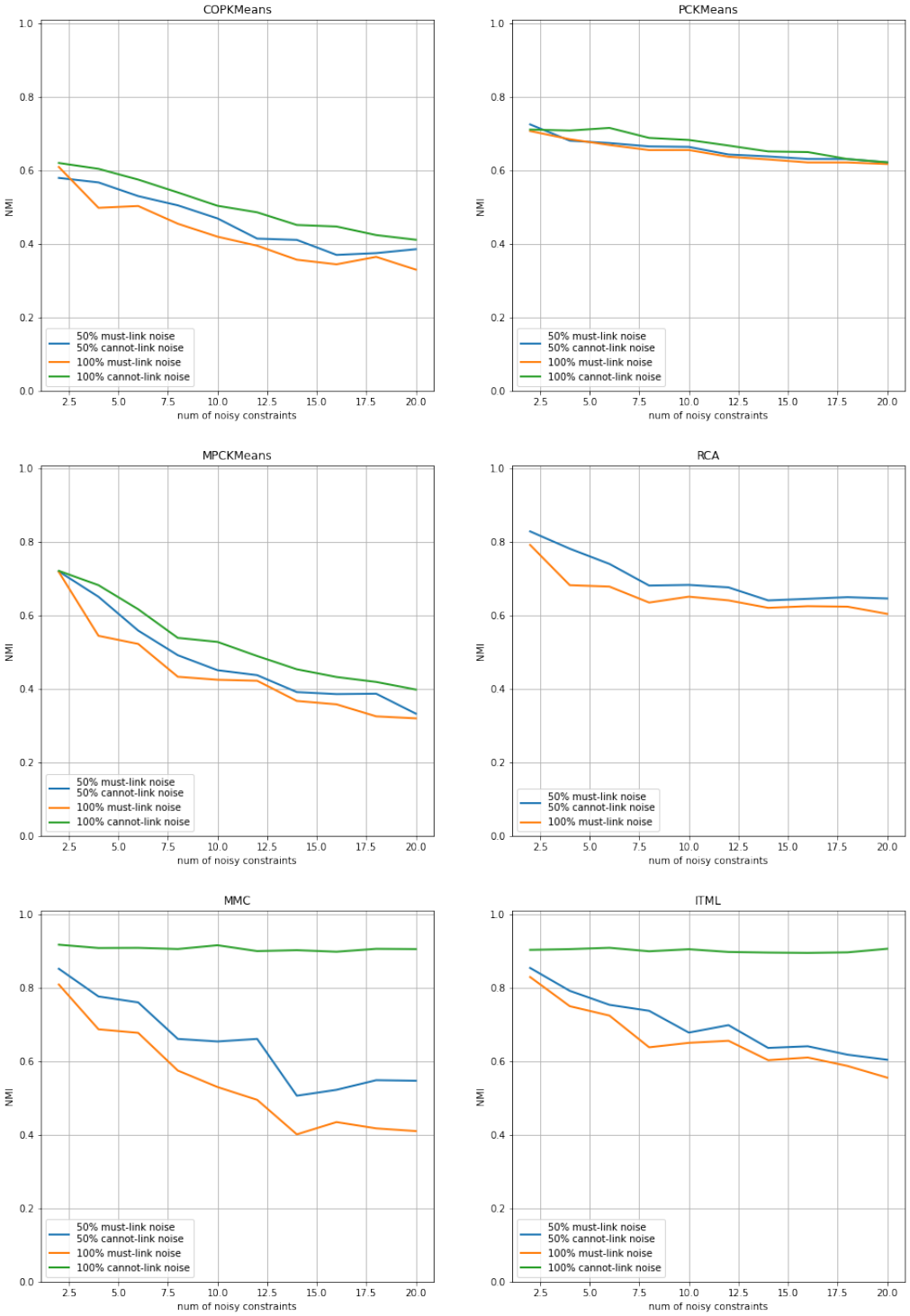
Among all metric learning methods, the RCA approach generally has the best performance, even if it only uses must-link constraints. MPC-Kmeans has the highest NMI index score, compared with other K-means variants.

8.4.2 Impact of Noisy Constraint Sets on Semi-supervised Clustering Approaches

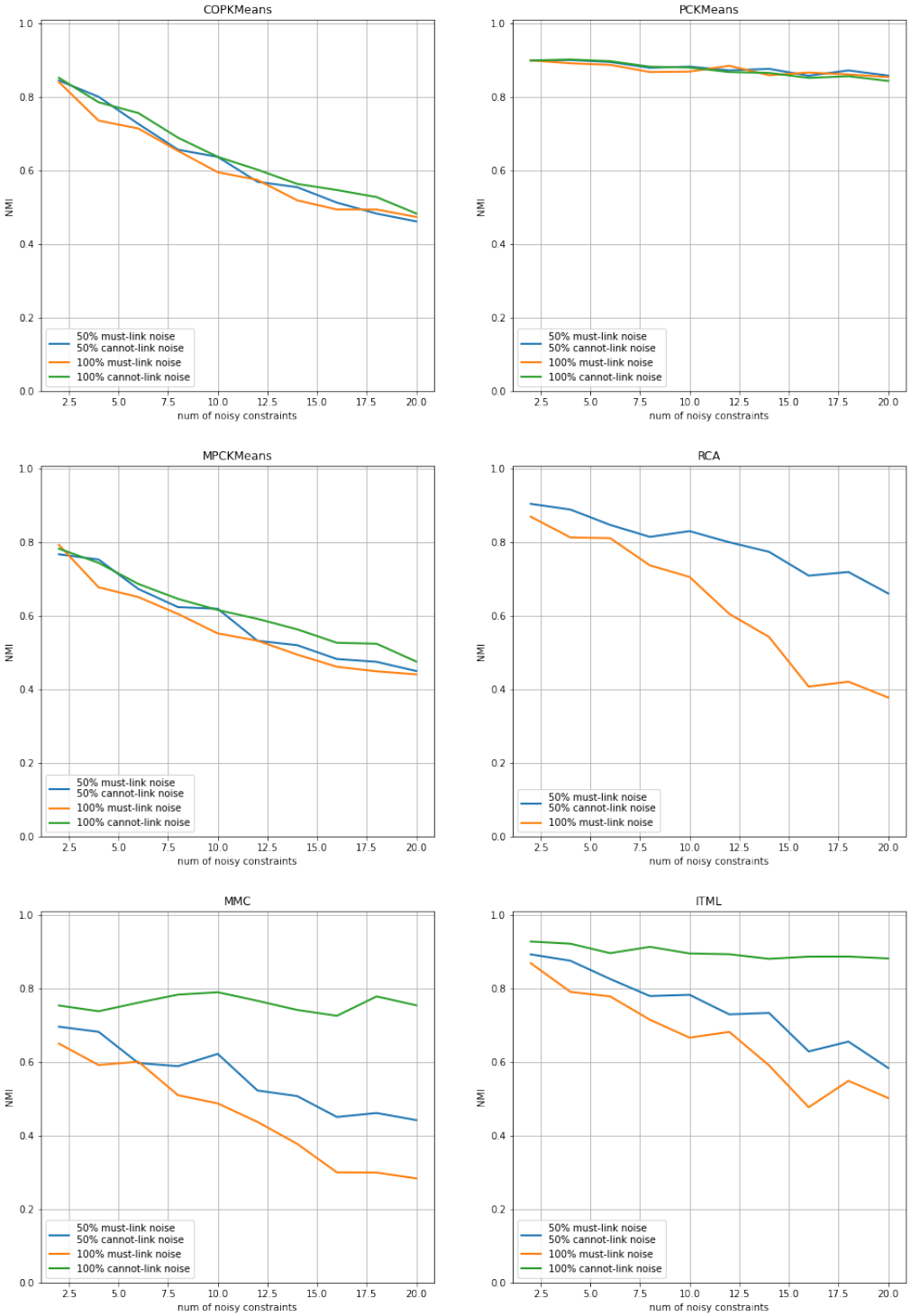
In this experiment, the impact of noisy constraint on the semi-supervised clustering approaches is analysed. The number of total pairwise constraints is fixed to 100, including 50 must-link constraints and 50 cannot-link constraints, guaranteeing that the semi-supervised clustering approaches have sufficient supervised information to generate initial clustering solution. The number of noisy constraints ranges from 0 to 20. For each number of noisy constraints, 30 different constraint sets were generated to get repeated trials. The generated noisy constraints may exist equally in must-link constraints and cannot-link constraints, or only in must-link constraints, or only in cannot-link constraints.

Figure 8.2 demonstrates the results of COP-Kmeans, PC-Kmeans, MPC-Kmeans, RCA, MMC and ITML for the Iris (a), Wine (b) and Seeds (c) datasets. The horizontal axis shows the number of total noisy constraints used during the run and the vertical axis shows the average NMI index score of the output clustering solution over all trials for each approach. The vertical axis is normalized to 0-1 range for all subplots in the figure. The blue curve corresponds to the NMI index score of each approach under the case that the number of noisy must-link constraints equals to the number of noisy cannot-link constraints. The orange curve and green curve respectively represent the NMI evaluation of each approach in the situation where the noisy constraints exist only in must-link constraints or only in cannot-link constraints.

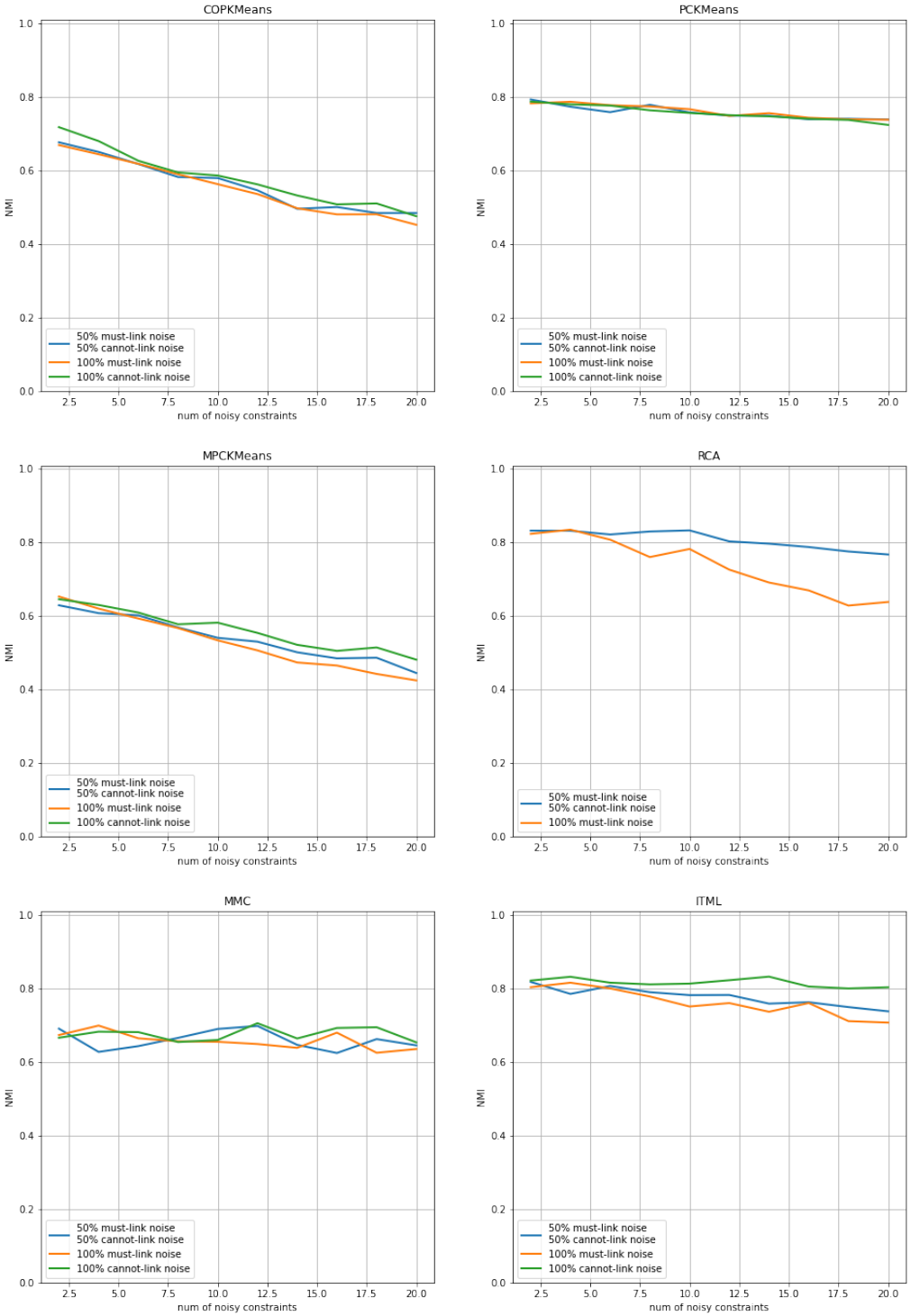
According to the analysis in the previous experiment, semi-supervised clustering approaches benefit more from must-link constraints. The finding is confirmed in this experiment, as we can observe from the orange curve, there is an important decrease on performance when noise exists in in must-link constraints.


COPKMeans
PCKMeans
MPCKMeans
RCA
MMC
ITML
NMI
num of noisy constraints
50% must-link noise
50% cannot-link noise
100% must-link noise
100% cannot-link noise

(a) Iris


COPKMeans
PCKMeans
MPCKMeans
RCA
MMC
ITML
NMI
num of noisy constraints
50% must-link noise
50% cannot-link noise
100% must-link noise
100% cannot-link noise

(b) Wine



(c) Seeds

Figure 8.2: Performance of semi-supervised clustering with noisy constraint sets.

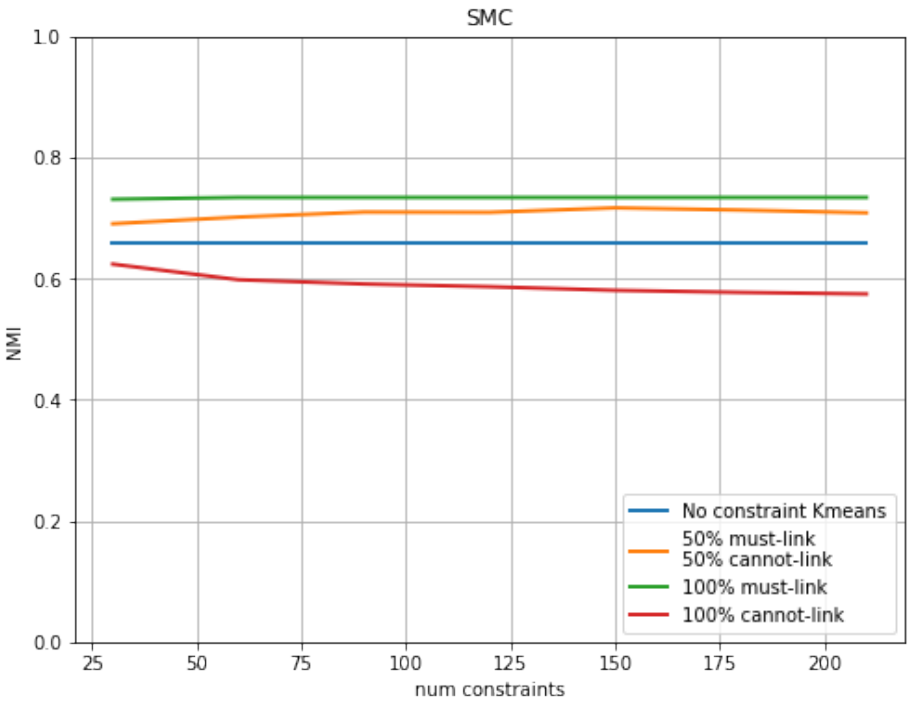
The MMC approach and the ITML approach have better robustness against noise in cannot-link constraints, as illustrated by the green curve. Except that, for all the other approaches, noisy constraints usually lead to a significant drop on performance, by comparing starting point and end point of the curves, showing the weakness of these approaches in stability, especially when noisy constraint exists in must-link constraints. The good robustness of the MMC approach and the PC-Kmeans approach on Wine and Seeds datasets actually comes from the fact that the performance of the clustering solution without noise is poor, the noisy constraint therefore does not result in worse performance.

8.4.3 Impact of Unbalanced and Noisy Constraint Sets on Semi-MultiCons

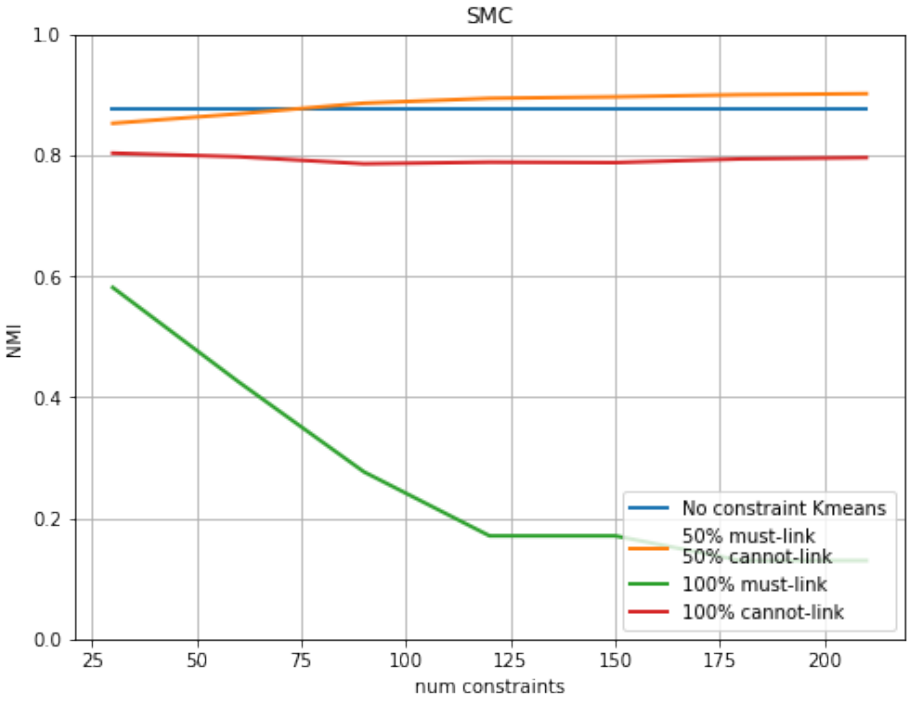
We conducted the same experiments, as for semi-supervised clustering approaches, on the Semi-MultiCons approach to present its performance and robustness facing noisy input. The results of unbalanced constraint set for the Iris, Wine and Seeds datasets are presented in Figure 8.3. The blue curve corresponds to the unsupervised Kmeans clustering result when no constraint is used, with the input parameter k set to number of classes. The orange curve corresponds to S_r evaluation of the SMC approach in the case that the number of must-link constraints is equal to the number of cannot-link constraints. The green and red curves represent the S_r evaluation of the SMC approach in the situation where the constraint set contains respectively only must-link constraints and only cannot-link constraints.

Unlike semi-supervised clustering approaches, the SMC approach benefits more from cannot-link constraints than from must-link constraints, as presented by the red and green curves. This implies that the SMC approach makes use of both must-link constraints and cannot-link constraints to infer a proper number of clusters k. Indeed, the performance of the SMC approach is not as good as the single semi-supervised clustering approach when there is no noise in the constraint set; However, the SMC approach solves a more difficult problem that is the number of clusters k is unknown.

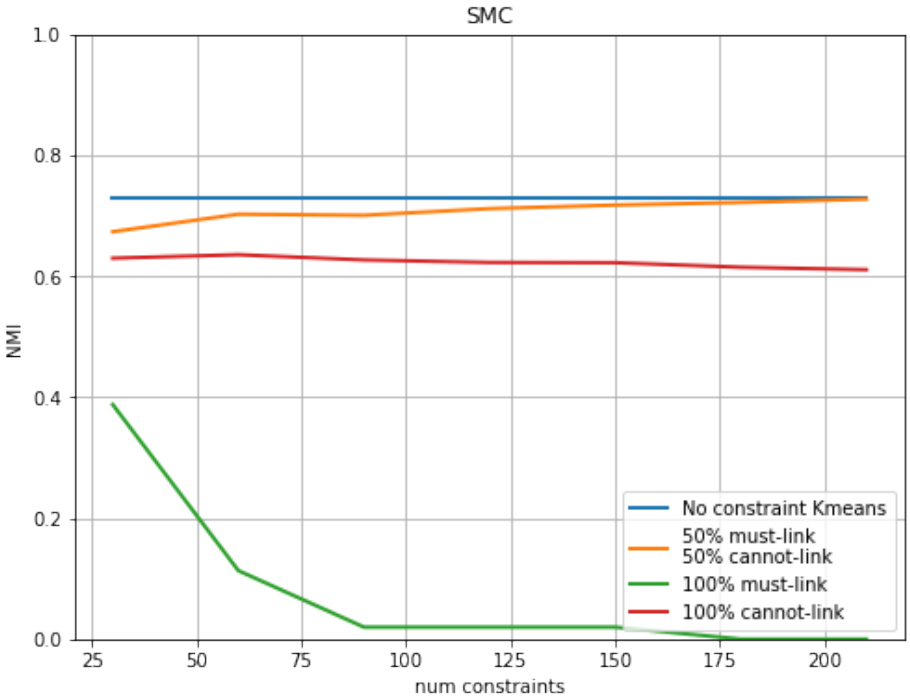
The results of noisy constraint sets for the Iris, Wine and Seeds datasets are presented in Figure 8.4. The blue curve corresponds to the S_r evaluation of the SMC approach in the case that the number of noisy must-link constraints equals to the number of noisy cannot-link constraints. The orange and green curves represent the S_r evaluation of the SMC approach in the situation where the noisy constraints exist respectively only among must-link constraints and only among cannot-link constraints.


SMC
NMI
num constraints
No constraint Kmeans
50% must-link
50% cannot-link
100% must-link
100% cannot-link

(a) Iris


SMC
NMI
num constraints
No constraint Kmeans
50% must-link
50% cannot-link
100% must-link
100% cannot-link

(b) Wine



(c) Seeds

Figure 8.3: Performance of Semi-MultiCons with unbalanced constraint sets. The impact of unbalanced constraint set on the Semi-MultiCons approach for the Iris (a), Wine (b) and Seeds (c) datasets are shown. The horizontal axis shows the number of total pairwise constraints used during the run and the vertical axis shows the average NMI index score of the output clustering solution over all trials. The vertical axis is normalized to [0.0, 1.0] range.

For all tested benchmark datasets, the SMC approach achieves remarkable robustness against both must-link noisy constraints and cannot-link noisy constraints, proving that integrating constraints in the consensus process provides a better stability than using semi-supervised clustering approaches.

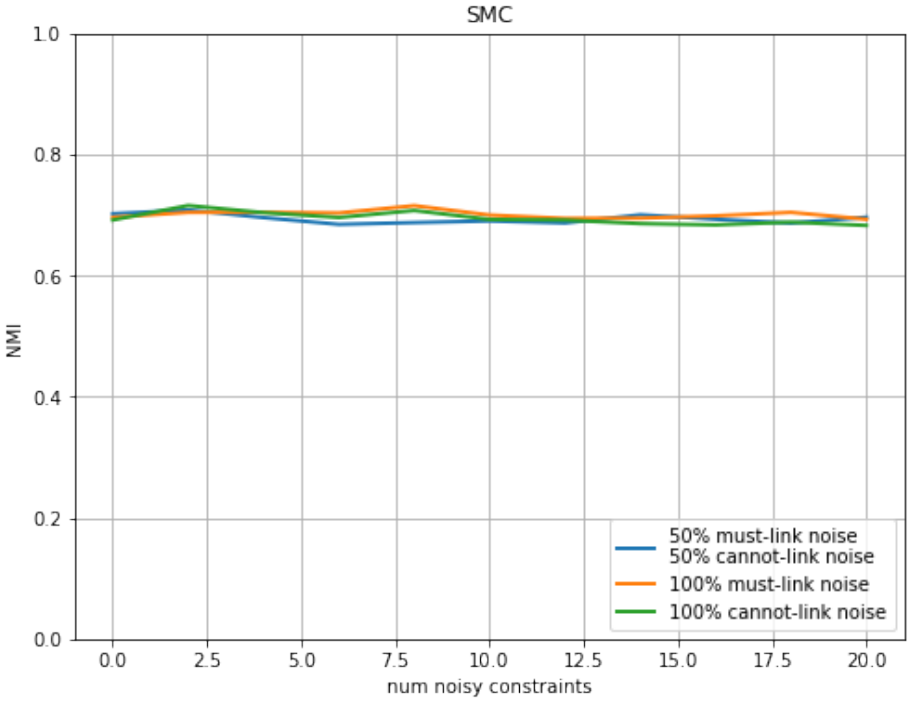
8.5 Conclusion

The experiments about the impact of unbalanced constraint sets and noisy constraint sets on semi-supervised clustering approaches and Semi-MultiCons have led to the following conclusions:

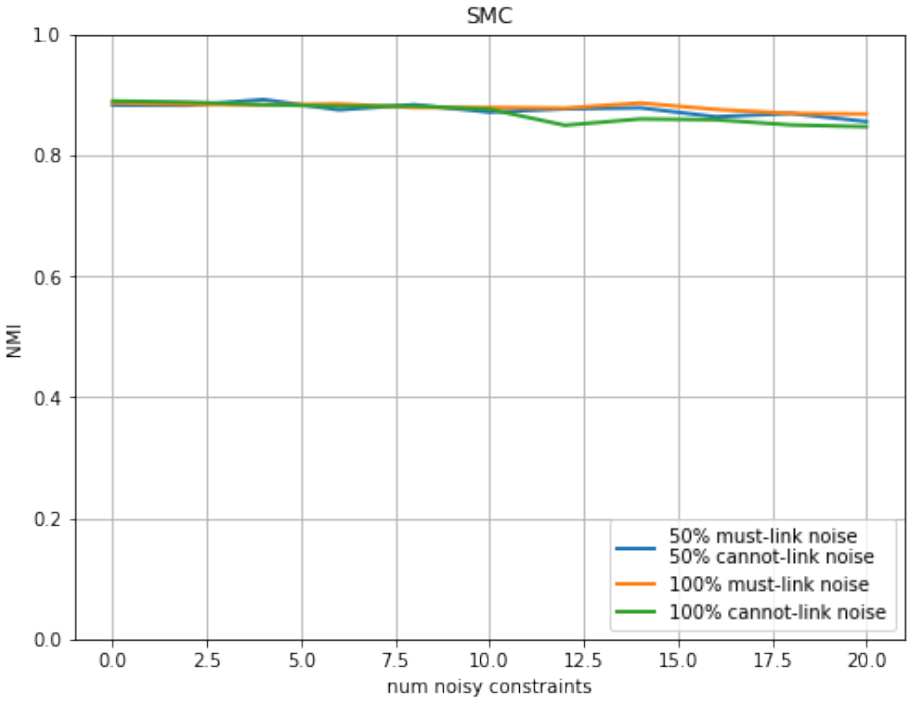
- The metric learning semi-supervised clustering approaches converge faster than semi-supervised clustering approaches that are K-means variants. However, K-means variants are able to achieve better performance when the number of constraints is large enough.
- Must-link constraints have a significant positive impact on the performance of semi-supervised clustering approaches. In contrast, using only cannot-link constraints usually leads to a decrease in performance.
- Most semi-supervised clustering approaches do not have a good robustness against noisy constraints, especially noisy must-link constraints.
- Semi-MultiCons is able to make use of both must-link constraints and cannot-link constraints, and requires both types of constraints to infer a proper number of clusters k.
- Semi-MultiCons achieves remarkable robustness against both must-link noisy constraints and cannot-link noisy constraints. However, the performance of Semi-MultiCons is not as good as the single semi-supervised clustering approach, which is a consequent of the fact that Semi-MultiCons solves the more challenging problem that the number of clusters k is unknown.

Based on these conclusions, we hereafter highlight the scenarios for which each approach is more suitable for:

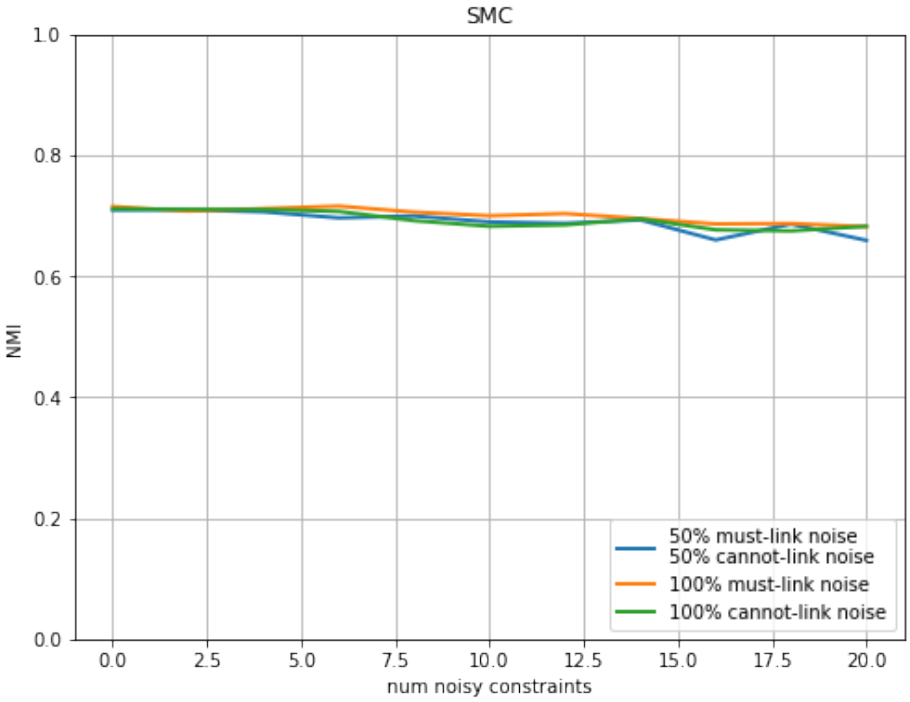
- If the number of constraints is small, it is suggested to use metric learn semi-supervised clustering approaches. In the case that the constraint set mainly contains must-link constraints, the RCA approach is proposed. Otherwise the MMC approach and the ITML approach are worth to test.


SMC
NMI
num noisy constraints
50% must-link noise
50% cannot-link noise
100% must-link noise
100% cannot-link noise

(a) Iris


SMC
NMI
num noisy constraints
50% must-link noise
50% cannot-link noise
100% must-link noise
100% cannot-link noise

(b) Wine



(c) Seeds

Figure 8.4: Performance of Semi-MultiCons with noisy constraint sets. The impact of noisy constraint set on the Semi-MultiCons approach for the Iris (a), Wine (b) and Seeds (c) datasets are presented. The horizontal axis shows the number of total noisy constraints used during the run and the vertical axis shows the average NMI index score of the output clustering solution over all trials. The vertical axis is normalized to [0.0, 1.0] range.

- If the number of constraints is large, it is suggested to use semi-supervised clustering approaches that are K-means variants, especially MPC-Kmeans.
- If the constraint set possibly contains noisy constraints, it is suggested to use the Semi-MultiCons approach to prevent significant drop in performance.

Chapter 9

Conclusion and Future Work

The Amadeus Revenue Accounting System automatically process tickets until an error occurs. The workflow is then interrupted and manual corrections of errors are required. Each error needs to be corrected by users and put back into the workflow. The workflow may then be interrupted again, following the same recovery process. The main problem here is that each error is treated as independent, even if similar errors have already been corrected. The analysis of a sample of 2 000 error correction tasks have shown up to 40% similar tasks, implying an important waste of efforts. The work conducted during this thesis aims to improve the automation of the error handling process through two steps: The first is the clustering of error tickets to model clusters of tickets corresponding to similar anomalies and requiring similar correction processes; The second is the assignment to error ticket clusters of correction actions performed by the users, and to make use of this supervision information to improve the clustering quality afterwards.

To achieve these goals and to address the central issues presented in Chapter 2 about applying classical clustering approaches to Revenue Accounting Workflow data, we proposed a new semi-supervised clustering approach, named Semi-MultiCons, in Chapter 4. Our contribution includes the development of a novel iterative constraint-based merging/splitting consensus function based on the initial MultiCons approach, the optimization of implementation to reduce its complexity, as well as the proposition of a new constraint-based consensus selection method.

To extensively evaluate the effect of integrating constraints in the ensemble member creation step and in the consensus generation step of Semi-MultiCons, four different Semi-MultiCons algorithms were implemented and their performances were analyzed in detail. The experimental results on UCI MLR benchmark datasets demonstrated in 6 proves that Semi-MultiCons 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 Semi-MultiCons approach was applied on real industrial dataset as well. Based on the experiments conducted on Amadeus datasets from three different customers and two task types, Semi-MultiCons was shown to be able to handle huge industrial datasets and to achieve good performance for all customers and task types. The proposed mini-batch mode notably improves the efficiency and the scalability of Semi-MultiCons, and makes it possible for Semi-MultiCons to give quick or even real-time response.

We created a real-life Proof-of-Concept prototype with Big Data ecosystem and Cloud platform to deploy Semi-MultiCons in a real industrial environment. With this PoC, the user is able to explore clusters of similar tasks in the Semi-MultiCons hierarchical clustering result, to visualize their feature analysis for understanding why tasks are grouped or separated, to validate clusters of similar tasks, as well as to make batch fixes or corrections per cluster. The validation data from users will then be used as supervised information to improve the overall quality of Semi-MultiCons clustering result later on.

We also analyzed the impact of unbalanced constraint sets and noisy constraint sets on the performance of both Semi-MultiCons and other semi-supervised clustering approaches. Unlike other semi-supervised clustering approaches that benefit mainly from must-link constraints, Semi-MultiCons is able to make use of both must-link constraints and cannot-link constraints, and it requires both types of constraints to infer a relevant number of clusters k. Moreover, Semi-MultiCons achieves remarkable robustness against both must-link noisy constraints and cannot-link noisy constraints.

Further work on the Semi-MultiCons approach involves the integration of pairwise constraints in the closed pattern mining phase, for generating constraints-based clustering patterns, the collection of real must-link and cannot-link constraints from users to obtain concrete results, and extended experiments on other customers and task types to fully investigate the performance of Semi-MultiCons on Amadeus data. We also plan to evaluate and adapt our PoC according to the feedback from the Revenue Accounting domain experts and Amadeus customers. The evaluation will focus in particular on the relevance of clusters of similar tasks, as well as the average resolution time per task in our PoC compared to the current Task Handling Module.

Bibliography

[1] Atheer Al-Najdi. *A closed Patterns-based Approach to the Consensus Clustering Problem*. PhD thesis, Université Côte d'Azur, 2016.

[2] Atheer Al-Najdi, Nicolas Pasquier, and Frédéric Precioso. Multiple consensuses clustering by iterative merging/splitting of clustering patterns. In *Proceedings of the International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, pages 790–804. Springer, 2016.

[3] Muna Al-Razgan and Carlotta Domeniconi. Clustering ensembles with active constraints. In *Applications of Supervised and Unsupervised Ensemble Methods*, pages 175–189. Springer, 2009.

[4] M Eduardo Ares, Javier Parapar, and Álvaro Barreiro. An experimental study of constrained clustering effectiveness in presence of erroneous constraints. *Information Processing & Management (IPM)*, 48(3):537–551, 2012.

[5] Aharon Bar-Hillel, Tomer Hertz, Noam Shental, Daphna Weinshall, and Greg Ridgeway. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research (JMLR)*, 6(6), 2005.

[6] Sugato Basu, Mikhail Bilenko, and Raymond J Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the 10th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 59–68. ACM, 2004.

[7] Karell Bertet, Christophe Demko, Jean-François Viaud, and Clément Guérin. Lattices, closures systems and implication bases: A survey of structural aspects and algorithms. *Theoretical Computer Science (TCS)*, 743:93–109, 2018.

[8] Mikhail Bilenko, Sugato Basu, and Raymond J Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the 21st International Conference on Machine learning (ICML)*, pages 81–88. ACM, 2004.

[9] Veselka Boeva, Milena Angelova, Niklas Lavesson, Oliver Rosander, and Elena Tsiporkova. Evolutionary clustering techniques for expertise mining scenarios. In *Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART)*, pages 523–530, 2018.

[10] Tossapon Boongoen and Natthakan Iam-On. Cluster ensembles: A survey of approaches with recent extensions and applications. *Computer Science Review*, 28:1–25, 2018.

[11] Shiguo Chen and Daoqiang Zhang. Semisupervised dimensionality reduction with pairwise constraints for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters (GRSL)*, 8(2):369–373, 2011.

[12] David Cohn, Rich Caruana, and Andrew McCallum. Semi-supervised clustering with user feedback. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, 4(1):17–32, 2003.

[13] Tom Coleman, James Saunderson, and Anthony Wirth. Spectral clustering with inconsistent advice. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 152–159, 2008.

[14] Thiago F Covoes, Eduardo R Hruschka, and Joydeep Ghosh. A study of k-means-based algorithms for constrained clustering. *Intelligent Data Analysis (IDA)*, 17(3):485–505, 2013.

[15] Lori Dalton, Virginia Ballarin, and Marcel Brun. Clustering algorithms: On learning, validation, performance, and applications to genomics. *Current Genomics*, 10(6):430–445, 2009.

[16] Ian Davidson and Sugato Basu. A survey of clustering with instance level. *Constraints*, 1:2, 2007.

[17] Ian Davidson and SS Ravi. Clustering with constraints: Feasibility issues and the k-means algorithm. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 138–149. SIAM, 2005.

[18] Ian Davidson and SS Ravi. Identifying and generating easy sets of constraints for clustering. In *AAAI*, volume 6, page 336341, 2006.

[19] Ian Davidson and SS Ravi. The complexity of non-hierarchical clustering with instance and cluster level constraints. *Data Mining and Knowledge Discovery (DMKD)*, 14(1):25–61, 2007.

[20] Ian Davidson, SS Ravi, and Leonid Shamis. A sat-based framework for efficient constrained clustering. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 94–105. SIAM, 2010.

[21] Ian Davidson, Kiri L Wagstaff, and Sugato Basu. Measuring constraint-set utility for partitional clustering algorithms. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 115–126. Springer, 2006.

[22] Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 209–216, 2007.

[23] William de Vazelhes, CJ Carey, Yuan Tang, Nathalie Vauquier, and Aurélien Bellet. metric-learn: Metric Learning Algorithms in Python. *Journal of Machine Learning Research (JMLR)*, 21(138):1–6, 2020.

[24] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 16(07):901–912, 2002.

[25] Derya Dinler and Mustafa Kemal Tural. A survey of constrained clustering. In *Unsupervised Learning Algorithms*, pages 207–235. Springer, 2016.

[26] Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017. http://archive.ics.uci.edu/ml.

[27] Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y Zomaya, Sebti Foufou, and Abdelaziz Bouras. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2(3):267–279, 2014.

[28] Ines Färber, Stephan Günnemann, Hans-Peter Kriegel, Peer Kröger, Emmanuel Müller, Erich Schubert, Thomas Seidl, and Arthur Zimek. On using class-labels in evaluation of cluster-

ings. In *MultiClust: 1st International Workshop on Discovering, Summarizing and using Multiple Clusterings held in conjunction with KDD*, page 1, 2010.

[29] Xiaoli Zhang Fern and Carla E Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the 21st International Conference on Machine learning (ICML)*, page 36. ACM, 2004.

[30] Germain Forestier, Pierre Gançarski, and Cédric Wemmert. Collaborative clustering with background knowledge. *Data & Knowledge Engineering (DKE)*, 69(2):211–228, 2010.

[31] Pierre Gançarski, Bruno Crémilleux, Germain Forestier, Thomas Lampert, et al. Constrained clustering: Current and new trends. In *A Guided Tour of Artificial Intelligence Research*, pages 447–484. Springer, 2020.

[32] Mohadeseh Ganji, James Bailey, and Peter J Stuckey. Lagrangian constrained clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining (SDM)*, pages 288–296. SIAM, 2016.

[33] Sean Gilpin and Ian Davidson. A flexible ilp formulation for hierarchical clustering. *Artificial Intelligence*, 244:95–109, 2017.

[34] AD Gordon and M Vichi. Fuzzy partition models for fitting a set of partitions. *Psychometrika*, 66(2):229–247, 2001.

[35] Tias Guns, Siegfried Nijssen, and Luc De Raedt. k-pattern set mining under constraints. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 25(2):402–418, 2011.

[36] Michael Hahsler, Sudheer Chelluboina, Kurt Hornik, and Christian Buchta. The arules r-package ecosystem: analyzing interesting patterns from large transaction data sets. *Journal of Machine Learning Research (JMLR)*, 12(Jun):2021–2025, 2011.

[37] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems (JIIS)*, 17(2):107–145, 2001.

[38] Christian Hennig. Clustering strategy and method selection. *arXiv preprint arXiv:1503.02059*, 2015.

[39] Steven CH Hoi, Rong Jin, and Michael R Lyu. Learning nonparametric kernel matrices from pairwise constraints. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 361–368. ACM, 2007.

[40] Kurt Hornik. A clue for cluster ensembles. *Journal of Statistical Software*, 14(12):1–25, 2005.

[41] Kurt Hornik and Walter Böhm. Hard and soft euclidean consensus partitions. In *Data Analysis, Machine Learning and Applications*, pages 147–154. Springer, 2008.

[42] Ashraf Mohammed Iqbal, Abidalrahman Moh'd, and Zahoor Khan. Semi-supervised clustering ensemble by voting. *arXiv preprint arXiv:1208.4138*, 2012.

[43] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New Phytologist*, 11(2):37–50, 1912.

[44] Ayan Acharya Joydeep Ghosh. *A Survey of Consensus Clustering*, chapter 22. CRC Press, 2015.

[45] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

[46] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(1):1–58, 2009.

[47] Brian Kulis et al. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.

[48] Tarald Kvålseth. On normalized mutual information: Measure derivations and properties. *Entropy*, 19(11):631, 2017.

[49] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database, 2010.

[50] Zhiwu Lu and Yuxin Peng. Exhaustive and efficient constraint propagation: A graph-based learning approach and its applications. *International Journal of Computer Vision (IJCV)*, 103(3):306–325, 2013.

[51] Amjad Mahmood, Tianrui Li, Yan Yang, Hongjun Wang, and Mehtab Afzal. Semi-supervised evolutionary ensembles for web video categorization. *Knowledge-Based Systems (KBS)*, 76:53–66, 2015.

[52] K. C. Mondal, N. Pasquier, A. Mukhopadhyay, U. Maulik, and S. Bandhopadyay. A new approach for association rule mining and bi-clustering using formal concept analysis. In *Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, volume 7376 of *Lecture Notes in Computer Science*, pages 86–101. Springer, 2012.

[53] Blaine Nelson and Ira Cohen. Revisiting probabilistic models for clustering with pair-wise constraints. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 673–680, 2007.

[54] Abdelkader Ouali, Samir Loudni, Yahia Lebbah, Patrice Boizumault, Albrecht Zimmermann, and Lakhdar Loukil. Efficiently finding conceptual clustering models with integer linear programming. In *Proceedings of the 25th International Joint Conferences on Artificial Intelligence (IJCAI)*, 2016.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)*, 12:2825–2830, 2011.

[56] Dan Pelleg and Dorit Baras. K-means with large and noisy constraint sets. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 674–682. Springer, 2007.

[57] Ruggero Gaetano Pensa, Céline Robardet, Jean-François Boulicaut, et al. Constraint-driven co-clustering of 0/1 data. *Constrained Clustering: Advances in Algorithms, Theory and Applications*, pages 123–148, 2008.

[58] Guo-Jun Qi, Jinhui Tang, Zheng-Jun Zha, Tat-Seng Chua, and Hong-Jiang Zhang. An efficient sparse metric learning in high-dimensional space via l 1-penalized log-determinant regularization. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pages 841–848, 2009.

[59] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019.

[60] Eréndira Rendón, Itzel Abundez, Alejandra Arizmendi, and Elvia M Quiroz. Internal versus external cluster validation indexes. *International Journal of Computers and Communications*, 5(1):27–34, 2011.

[61] Carlos Ruiz, Myra Spiliopoulou, and Ernestina Menasalvas. Density-based semi-supervised clustering. *Data Mining and Knowledge Discovery (DMKD)*, 21(3):345–370, 2010.

[62] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

[63] Noam Shental, Tomer Hertz, Daphna Weinshall, and Misha Pavel. Adjustment learning and relevant component analysis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 776–790. Springer, 2002.

[64] Lindsay I. Smith. A Tutorial on Principal Components Analysis. Technical report, University of Otago, New Zealand, 2002.

[65] Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research (JMLR)*, 3(Dec):583–617, 2002.

[66] Jakub Svehla. Active semi-supervised clustering. *https://github.com/datamole-ai/active-semi-supervised-clustering*, 2020.

[67] Wei Tang, Hui Xiong, Shi Zhong, and Jie Wu. Enhancing semi-supervised clustering: a feature projection perspective. In *Proceedings of the 13th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 707–716. ACM, 2007.

[68] Caroline Tomasini, Eduardo Nunes Borges, Karina S Machado, and Leonardo R Emmendorfer. A study on the relationship between internal and external validity indices applied to partitioning and density-based clustering algorithms. In *ICEIS (1)*, pages 89–98, 2017.

[69] Alexander Topchy, Anil K Jain, and William Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1866–1881, 2005.

[70] Alexander P Topchy, Martin H. C. Law, Anil K. Jain, and Ana L. N. Fred. Analysis of consensus partition in cluster ensemble. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)*, pages 225–232. IEEE, 2004.

[71] Nguyen Minh Duc Tran Khanh Hiep. *conclust: Pairwise Constraints Clustering*, 2016. R package version 1.1.

[72] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, 9(11), 2008.

[73] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011.

[74] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. *AAAI/IAAI*, 1097:577–584, 2000.

[75] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 577–584, 2001.

[76] Kiri L Wagstaff. Value, cost, and sharing: Open issues in constrained clustering. In *International Workshop on Knowledge Discovery in Inductive Databases (KDID)*, pages 1–10. Springer, 2006.

[77] Kiri L Wagstaff, Sugato Basu, and Ian Davidson. When is constrained clustering beneficial, and why? *Ionosphere*, 58(60.1):62–63, 2006.

[78] Kiri Lou Wagstaff and Claire Cardie. *Intelligent clustering with instance-level constraints*. Cornell University USA, 2002.

[79] Fei Wang. Semisupervised metric learning by maximizing constraint margin. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(4):931–939, 2011.

[80] Yunli Wang and Youlian Pan. Semi-supervised consensus clustering for gene expression data analysis. *BioData Mining*, 7(1):7, 2014.

[81] Junjie Wu, Hui Xiong, and Jian Chen. Adapting the right measures for k-means clustering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 877–886, 2009.

[82] Wenchao Xiao, Yan Yang, Hongjun Wang, Tianrui Li, and Huanlai Xing. Semi-supervised hierarchical clustering ensemble and its application. *Neurocomputing*, 173:1362–1376, 2016.

[83] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, pages 521–528, 2003.

[84] Eric P Xing, Andrew Y Ng, Michael I Jordan, and Stuart Russell. Distance metric learning with application to clustering with side-information. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, page 521–528. MIT Press, 2002.

[85] Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.

[86] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

[87] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th ACM SIGIR International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 267–273, 2003.

[88] S. Ben Yahia, T. Hamrouni, and E. Mephu Nguifo. Frequent closed itemset based algorithms: A thorough structural and analytical survey. *SIGKDD Explorations Newsletter*, 8(1):93–104, 2006.

[89] Liu Yang and Rong Jin. *Distance Metric Learning: A Comprehensive Survey*. Michigan State University, 2006.

[90] Tianshu Yang. Implementation of the SMC Approach for Semi-supervised Consensus Clustering Based on Closed Patterns, 2021. *https://github.com/lazzyCloud/semi-multicons*.

[91] Tianshu Yang, Nicolas Pasquier, Antoine Hom, Laurent Dollé, and Frédéric Precioso. Semi-supervised Consensus Clustering Based on Frequent Closed Itemsets. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)* (Acceptance Rate 18%), pages 3341–3344. ACM, October 2020. Registered as Defensive

Paper of the Intellectual Property Invention "Clustering Techniques for Revenue Accounting Error-Handling Automation", Patent number ID2326WW00, Intellectual Property Board, Amadeus S.A.S., Sophia Antipolis, France. July, 2020.

[92] Tianshu Yang, Nicolas Pasquier, and Frédéric Precioso. Ensemble Clustering Based Semi-Supervised Learning for Revenue Accounting Workflow Management. In *Proceedings of the 9th International Conference on Data Science, Technology and Applications (DATA)* (Acceptance Rate 14%), pages 283–293. SciTePress, July 2020. Registered as Defensive Paper of the Intellectual Property Invention "Clustering Techniques for Revenue Accounting Error-Handling Automation", Patent number ID2326WW00, Intellectual Property Board, Amadeus S.A.S., Sophia Antipolis, France. July, 2020.

[93] Tianshu Yang, Nicolas Pasquier, and Frédéric Precioso. Semi-supervised consensus clustering based on frequent closed itemsets. In *PhD Track of the 18th International Symposium on Intelligent Data Analysis (IDA)*, Bodenseeforum, Germany, April 2020. Track cancelled due to the COVID-19 pandemic.

[94] Tianshu Yang, Nicolas Pasquier, and Frédéric Precioso. Semi-supervised consensus clustering based on closed patterns. *Knowledge-Based Systems (KBS)* (Impact Factor 8.038), 235(107599), January 2022.

[95] Tianshu Yang, Nicolas Pasquier, Frédéric Precioso, Antoine Hom, and Laurent Dollé. Clustering Techniques for Revenue Accounting Error-Handling Automation. *Intellectual Property Invention Patent ID2326WW00*, Intellectual Property Board, Amadeus S.A.S., Sophia Antipolis, France, July 2020.

[96] Yan Yang, Wei Tan, Tianrui Li, and Da Ruan. Consensus clustering based on constrained self-organizing map and improved cop-kmeans ensemble in intelligent decision support systems. *Knowledge-Based Systems (KBS)*, 32:101–115, 2012.

[97] Zhiwen Yu, Peinan Luo, Jiming Liu, Hau-San Wong, Jane You, Guoqiang Han, and Jun Zhang. Semi-supervised ensemble clustering based on selected constraint projection. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 30(12):2394–2407, 2018.

[98] Zhiwen Yu, Peinan Luo, Jane You, Hau-San Wong, Hareton Leung, Si Wu, Jun Zhang, and Guoqiang Han. Incremental semi-supervised clustering ensemble for high dimensional data

clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 28(3):701–714, 2016.

[99] Zhiwen Yu, Hau-San Wongb, Jane You, Qinmin Yang, and Hongying Liao. Knowledge based cluster ensemble for cancer discovery from biomolecular data. *IEEE Transactions on Nanobioscience*, 10(2):76–85, 2011.

[100] Hongjing Zhang, Sugato Basu, and Ian Davidson. Deep constrained clustering-algorithms and advances. *arXiv preprint arXiv:1901.10061*, 2019.

[101] Hongjing Zhang, Sugato Basu, and Ian Davidson. A framework for deep constrained clustering-algorithms and advances. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 57–72. Springer, 2019.

[102] Hongjing Zhang, Tianyang Zhan, Sugato Basu, and Ian Davidson. A framework for deep constrained clustering. *Data Mining and Knowledge Discovery (DMKD)*, pages 1–28, 2021.