6

ERGEBNISSE AUS DER INFORMATIK

The technological advancements of recent years led to a pervasion of all life areas with information systems and allows to conveniently and affordably gather large amounts of data. The key to our information society is the transformation of the mere data in these comprehensive databases into information and knowledge. One research area committed to this goal is the one of data mining, where the task is to automatically or semi-automatically extract previously unknown patterns from such data sources. The subject of this thesis is the mining task of clustering, which aims at grouping objects based on their similarity such that similar objects are grouped together, while dissimilar ones are separated.

Since modern storage systems are not subject to practical limitations anymore, data can be captured in its full complexity without restriction to a small selective set of aspects. For such complex data, just identifying a single clustering is often not sufficient. Instead, multiple, alternative, and valid clusterings can be identified for a single dataset, each highlighting different aspects of the data. The paradigm of *multi-view clustering*, also referred to as *alternative clustering*, is dedicated to explicitly discover such a diverse set of multiple, alternative clusterings in order to find all hidden patterns in the data.

A second observation for complex data sources, where usually many characteristics are stored for each object, is the inability to find similar objects by considering all of these characteristics. While clustering based on all attributes, in the full-space, is futile, valuable cluster patterns can be found for subsets of attributes, in subspace projections. This problem is tackled by approaches of the *subspace clustering* paradigm, which aim at uncovering clustering structures hidden in subspace projections, such that for each cluster a set of relevant attributes is determined automatically.

In this thesis, we want to highlight fundamental parallels between the two paradigms of multi-view clustering and subspace clustering, since both account for the possibility of objects belonging to multiple clusters simultaneously. Consequently, we present several approaches exploiting synergy effects by combining both paradigms to find multiple, alternative clusterings in subspace projections of the data.

ISBN 978-3-86359-368-1

9 783863 593681

Alternative Clustering in Subspace Projections

Ines Färber

Ines Färber

Alternative Clustering in Subspace Projections

Edition Wissenschaft Apprimus

Alternative Clustering in Subspace Projections

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Diplom-Informatikerin
Ines Färber
aus Jena

Berichter: Universitätsprofessor Dr. rer. nat. Thomas Seidl
Universitätsprofessorin Dr. rer. nat. Ira Assent

Tag der mündlichen Prüfung: 04.12.2014

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Ines Färber:

Alternative Clustering in Subspace Projections

1. Auflage, 2015

Gedruckt auf holz- und säurefreiem Papier, 100% chlorfrei gebleicht.

Apprimus Verlag, Aachen, 2015
Wissenschaftsverlag des Instituts für Industriekommunikation und Fachmedien
an der RWTH Aachen
Steinbachstr. 25, 52074 Aachen
Internet: www.apprimus-verlag.de, E-Mail: info@apprimus-verlag.de

Printed in Germany

ISBN 978-3-86359-368-1

D 82 (Diss. RWTH Aachen University, 2014)

Acknowledgements

I want to thank all my mentors, colleagues, collaborators, friends, and family who have encouraged, guided, and inspired me these last years. I enjoy to think back to many creative discussions, amazing conference trips, nightly working sessions with lots of pizza, and the many won friendships.

My first and foremost thanks go to my advisor, Prof. Thomas Seidl, who has stirred my interest in data mining and was always available to give advice and encouragement. I am very grateful for the opportunity to work in his group; a place of great individual freedom, noncompeting collaboration, and inspiring discussions concerning a wide area of data mining questions.

I also want to thank my second advisor, Prof. Ira Assent. I very much appreciate her effort in reviewing this document and her valuable feedback.

The ideas and algorithms of this thesis mostly have been developed in a team. My special thanks go to Sergej Fries (for his deep friendship and his collaboration in the funded projects BioKeyS and Theseus), Brigitte Boden (for her collaboration on graph topics and the warm atmosphere in our joint office), Emmanuel Müller (for his pathbreaking guidance in the beginning of my research), Stephan Günnemann (for his tireless creativity), and Hardy Kremer (for sometimes putting things into the right perpective), who all had a major influence on my research. I also highly value the inspiring discussions with the other team members: Anca Zimmer, Marwan Hassani, Christian Beecks, Philipp Kranen, Philip Driessen, Roland Assam, Seran Uysal. They all provided a very friendly and enjoyable atmosphere. The realization of many ideas has only been possible with the help of our motivated and skilled thesis students and student assistants. Although not everything made it in this thesis, I want to thank Simon Wollwaage, Matthias Hannen, Patrick Gerwert, Sebastian Raubach, Grzegorz Stepien, Thomas Mausbach, Matthias Rüdiger, Kittipat Virochsiri, and Tamer Alkhouli.

I also gratefully remember the collaborations with other research groups during diverse projects, and especially name Andrada Tatu, Michael Hund, Tobias Schreck, Enrico Bertini, Prof. Oliver Deussen, Prof. Daniel Keim, Annahita Oswald, Bianca Wackersreuther, and Prof. Christian Böhm.

My sincerest gratitude I want to express to my family. I will be forever grateful for your unconditional love and support. Mom, I do not know who I would be without you. Flo, thanks for keeping me grounded in stressful times and for proofreading all this. BO, thank you for patiently letting me finish this thesis.

During the work for this thesis I received financial support from the German Research Society (DFG) under the research grant DFG-611 within the DFG Priority Program “Scalable Visual Analytics: Interactive Visual Analysis Systems of Complex Information Spaces” (SPP 1335), from the Federal Ministry for Economic Affairs and Energy (BMWi) under the Research Program “THESEUS Mittelstand - Machinet - Machine Intelligence Network”, and for the project “BioKeyS” by the Federal Office for Information Security (BSI).

Contents

Abstract

The technological advancements of recent years led to a pervasion of all life areas with information systems and allows to conveniently and affordably gather large amounts of data. The key to our information society is the transformation of the mere data in these comprehensive databases into information and knowledge. One research area committed to this goal is the one of data mining, where the task is to automatically or semi-automatically extract previously unknown patterns from such data sources. The subject of this thesis is the mining task of clustering, which aims at grouping objects based on their similarity such that similar objects are grouped together, while dissimilar ones are separated.

Since modern storage systems are not subject to practical limitations anymore, data can be captured in its full complexity without restriction to a small selective set of aspects. For such complex data, just identifying a single clustering is often not sufficient. Instead, multiple, alternative, and valid clusterings can be identified for a single dataset, each highlighting different aspects of the data. The paradigm of *multi-view clustering*, also referred to as *alternative clustering*, is dedicated to explicitly discover such a diverse set of multiple, alternative clusterings in order to find all hidden patterns in the data.

A second observation for complex data sources, where usually many characteristics are stored for each object, is the inability to find similar objects by considering all of these characteristics. While clustering based on all attributes, in the full-space, is futile, valuable cluster patterns can be found for subsets of attributes, in subspace projections. This problem is tackled by approaches of the *subspace clustering* paradigm, which aim at uncovering clustering structures hidden in subspace projections, such that for each cluster a set of relevant attributes is determined automatically.

In this thesis, we want to highlight fundamental parallels between the two paradigms of multi-view clustering and subspace clustering, since both account for the possibility of objects belonging to multiple clusters simultaneously. Consequently, we present several approaches exploiting synergy effects by combining both paradigms to find multiple, alternative clusterings in subspace projections of the data.

Zusammenfassung

Der bisherige technologische Fortschritt führte zu einer Durchdringung aller Lebensbereiche mit Informationssystemen und ermöglicht das einfache und günstige Erfassen großer Datenmengen. Für unsere Informationsgesellschaft ist es jedoch entscheidend aus diesen reichhaltigen Datenquellen nützliche Informationen und Wissen zu generieren. Diesem Ziel hat sich der Forschungsbereich des Data Mining gewidmet, dessen Aufgabe es ist automatisiert oder semiautomatisiert vorher unbekannte Muster aus Daten zu extrahieren. Diese Arbeit beschäftigt sich mit der Aufgabe des Clusterings, welche Objekte anhand ihrer Ähnlichkeit gruppiert.

Da moderne Speichertechnologien keine ernsthaften Grenzen mehr aufzeigen, können Daten meist in ihrer vollen Komplexität ohne eine Beschränkung auf lediglich ausgewählte Aspekte erfasst werden. Für solch komplexe Daten stellt jedoch ein einziges Clustering oft keine ausreichende Charakterisierung dar. Stattdessen lassen sich für einen Datensatz oft mehrere, unterschiedliche und sinnvolle Clusterings identifizieren. Das Paradigma des *Multi-View Clusterings*, auch als *Alternative Clustering* bezeichnet, hat sich dem Ziel verschrieben explizit nach einer solch diversen Menge mehrerer, alternativer Clusterings zu suchen um alle versteckten Muster der Daten aufzudecken.

Eine zweite Beobachtung für komplexe Daten, bei welchen üblicherweise für jedes Objekt eine Vielzahl von Eigenschaften erfasst wurde, ist eine sehr schwach ausgeprägte Ähnlichkeit zwischen Objekten bei Berücksichtigung all ihrer Merkmalsausprägungen. Während ein Clustering unter Berücksichtigung aller Attribute nicht zielführend ist, lassen sich bei Betrachtung einzelner Attributteilmengen, d.h. in Teilraumprojektionen, durchaus sinnvolle Clusterstrukturen identifizieren. Dieser Problemstellung haben sich Ansätze des *Subspace Clustering* Paradigmas angenommen, welche Clusterstrukturen in Teilraumprojektionen identifizieren, sodass für jeden Cluster automatisch auch die Menge der relevanten Attribute bestimmt wird.

In dieser Arbeit wollen wir die grundsätzlichen Parallelen beider Paradigmen, Multi-View Clustering und Subspace Clustering, hervorheben, da beiden die Eigenschaft der gleichzeitigen Zugehörigkeit einzelner Objekte zu mehreren Clustern gemein ist. Entsprechend stellen wir verschiedene Ansätze vor die durch die Kombination beider Paradigmen Synergieeffekte nutzen um mehrere, verschiedene Gruppierungen in Teilraumprojektionen zu identifizieren.

Part I

Introduction

The opposite of a correct statement is a false statement. But the opposite of a profound truth may well be another profound truth.

NIELS BOHR

1 Introduction

THE digitalization of our society combined with the increasing potential of technologies for storing and collecting data leads to an explosive growth of data sources. Efficiently and reliably storing and managing data in such massive databases is just the first challenge accompanying this trend. To unfold the full potential of the gathered data, the mere data has to be transformed into useful information. In science, engineering, and economy, data analysis is nowadays a necessity and enables the discovery of valuable patterns, trends, or anomalies in the data. Given the vast amount of data, human capabilities for manual analysis are quickly overstrained which generates an urgent need for techniques to automatically analyze and evaluate the collected raw data.

The multidisciplinary research field of *data mining*, as an essential part of the process for *knowledge discovery in databases* (KDD) [HKP11], is devoted to develop automatic or semi-automatic algorithms for detecting previously unknown and useful patterns in the data. The KDD process is a sequence of several important steps, which can be processed iteratively until the discovered patterns and the resulting insights meet the user's requirements (cf. Fig. 1.1). Before data mining techniques can be applied, the collected raw data usually needs to be preprocessed. This step can include the integration of data from several sources into one big data warehouse. Often the data quality needs improvement through data cleaning techniques to treat missing values, data inconsistencies, or noise sustained during data acquisition. Given the trend of unrestrainedly collecting all available data without targeting a specific analysis question, it is often inevitable to confine the consideration to only a selection of the data relevant to the analysis task and also to transform the data such that according mining techniques can be applied successfully. Among the various data mining principles one or several techniques can be applied on the task relevant data in order to extract useful patterns and characteristics of the data or to determine predictive models

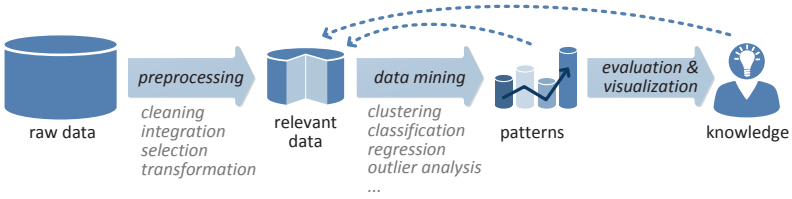


Figure 1.1: The knowledge discovery process in databases (KDD process)

to anticipate future trends. To help the user in interpreting and understanding the discovered patterns, the mining results are evaluated and visualized to translate the patterns into knowledge. Many data mining techniques involve user interaction, such that data mining and visualization are often coupled in a unifying, interactive framework. To influence or to enhance future mining results, the gathered knowledge as well as the discovered patterns of the KDD process can be integrated into a knowledge base as part of the relevant data for subsequent mining techniques.

In this thesis we will concentrate on the data mining, evaluation, and visualization steps of the KDD process with a special focus on data mining techniques. Furthermore we will consider the possibilities to utilize user knowledge or discovered patterns for subsequent mining processes. Out of the various data mining principles, we will restrict this thesis to the descriptive mining task of cluster analysis, which aims at grouping the data objects into clusters such that objects within each cluster are similar, while objects located in different clusters are dissimilar. Since clustering traditionally operates without given prior information, such as, e.g., partial information about class labels, but autonomously identifies the hidden aggregation structure of the data, it is ranked among the unsupervised mining principles. In contrast to clustering, the predictive mining task of classification needs partial information about the class structure in order to learn a model based on which unknown objects can be classified. As we will see in the context of this thesis, such a clear differentiation between unsupervised and supervised learning tasks does not apply to modern mining techniques anymore. Instead, different paradigms such as clustering and classification contribute to each other such that in both fields various so-called semi-supervised techniques exist. Also, the integration of previously detected patterns and user knowledge can be understood as semi-supervision as it is able to guide a clustering procedure in addition to the data itself. In this thesis, we will therefore consider unsupervised as well as semi-supervised clustering techniques.

Clustering analysis is widely studied in the data mining and machine learning community to detect patterns in the data, helping to identify structures and relationships in complex data as well as to summarize the data. Clustering techniques are also often applied as intermediate step for other data mining paradigms such as classification or outlier analysis, exploiting the summarizing characteristic of such a descriptive structural analysis. The research field of clustering is very diverse and the presented solutions highly depend on the targeted data domain (text, multimedia, networks, timeseries, numeric vector data, etc.) and the application's problem setting (streaming data, uncertain data, pattern type, segmentation, summarization, trend detection, etc.). In this thesis we will focus on numeric vector data but will also strive the topic of clustering within network data. Typical applications for those data domains can, for example, be :

- Customer segmentation: Here, the goal is to find groups of customers with similar buying, behavioral, or personal characteristics in order to find groups with common needs and priorities to enhance targeting and recommendation strategies.
- Sensor data analysis: By detecting sensor groups showing similar measurements, a compression of the data into cluster information can help in reducing the power consumption for long-distance transmissions of mobile sensors. It can also help to detect global events, trends, or anomalies.
- Gene expression data analysis: If represented as microarray, one goal is to find genes with homogeneous expression levels, which indicates that they share a common function. If gene interactions are additionally taken into account in a network representation, then a goal is to find genes that show similar expression levels and are densely connected to identify functional modules.
- Network analysis: Clusters in networks, also referred to as communities, are groups of densely connected vertices. Clusters in the World Wide Web, e.g., comprise web pages with topical similarities or identify link farms. In social networks, clusters correspond to social groups, e.g., different research divisions in a scientific co-authorship network.

Instead of just a small selection of relevant information, the grown potential of data storing and data recording techniques enables us to capture data from such different applications in its full complexity. For such complex data, traditional clustering methods are often incapable of detecting a meaningful or all-encompassing clustering structure. Therefore, several specialized areas have

formed within the research field of cluster analysis, among which we will concentrate on the one of multi-view or alternative clustering and on the one of subspace clustering.

1.1 Multi-View and Alternative Clustering

The process of clustering is commonly known to be very subjective and that there does not exist the one true clustering approach to solve all clustering problems. Instead many different cluster concepts exist, such as e.g., arbitrarily shaped clusters, compactness-based clusters, or distribution-based clusters. The clustering result, thus, strongly depends on the chosen clustering algorithm but also on the selected parameter setting for this algorithm. While the instability of clustering results has encouraged the research field of ensemble clustering to develop methods to find the one, unifying consensus clustering, the awareness of the inherent subjectivity of the clustering task motivated several researchers to go the opposite direction and explicitly uncover the diverse set of hidden clusterings within the data. The research area of alternative clustering, also referred to as multi-view clustering[1] follows the philosophy that some datasets, especially if they are complex, can allow for multiple valuable, alternative clusterings. Each one of these groupings has its justification and might present a reasonable view or perspective on the data's nature. A toy example that, in variations, is commonly used in the literature (e.g., [DQ08]) to visualize the validity of concurring alternative clusterings is depicted in Fig. 1.2. For such data, there is no indication to prefer the left over the right clustering as both partitionings fulfill similar quality requirements.

Besides the exploratory curiosity of scientists, we can find various reasons to extract all clusterings hiding in the considered data. A user often does not know in advance which data characterization is the most useful one for an application. In such a scenario the presentation of different available alternatives helps to evaluate the different options. For different applications, different clustering solutions might be suited best, such that a single clustering will not be sufficient. In some applications there might already exist a strong hypothesis on the clustering structure of the data and it is necessary to verify that there does not exist another strong, competing clustering structure.

[1] The term *multi-view clustering* is also commonly used for the clustering paradigm which searches for a single clustering of data represented by multiple different sources. For clarity will call this paradigm *multi-source clustering* in the remainder of this thesis.

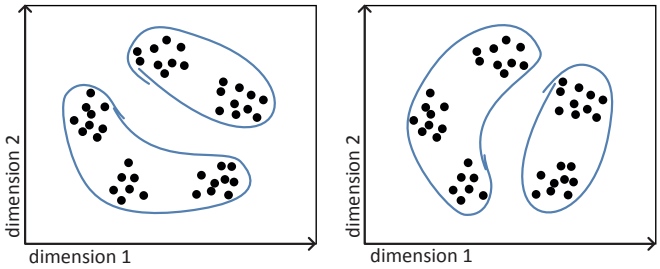


Figure 1.2: Alternative clusterings for a single dataset

1.2 Subspace Clustering

In our information society, we can observe the trend to collect all available information just in case of any potential future application. On the one hand, such rich data repositories represent treasures for data miners and might allow to reveal complex new patterns, such as, e.g., alternative clusterings. On the other hand, such unfiltered data sources possess the unfortunate characteristic that patterns are obfuscated by irrelevant information. Traditional clustering approaches consider the full attribute space to assess the similarity between objects, i.e., all of the objects' characteristics are taken into account. With an increasing number of characteristics, it becomes, however, more and more unlikely that two objects share similar values with respect to all attributes. Thus, we observe an increasing distance for an increasing dimensionality of the attribute space. While distance values grow with increasing dimensionality, the variance of the distances becomes nearly a constant. As a consequence, the discrimination power of distance functions deceases with increasing dimensionality of the data space, such that all objects seem equally similar. As an effect of this so-called "curse of dimensionality" [BGRS99], we can observe that in high-dimensional spaces nearest neighbor queries become instable and that it becomes increasingly difficult to estimate distributional parameters such as, e.g., the mean. The effects of the curse of dimensionality are especially strong if we have a high proportion of irrelevant features. Due to such irrelevant features, it is very unlikely for traditional clustering methods to discover reasonable clustering structures in the full-space.

One possible solution to diminish the effects in high-dimensional spaces are techniques for global dimensionality reduction, e.g., the Principal Component Analysis (PCA [Jol02]). All objects are projected into a single low-dimensional

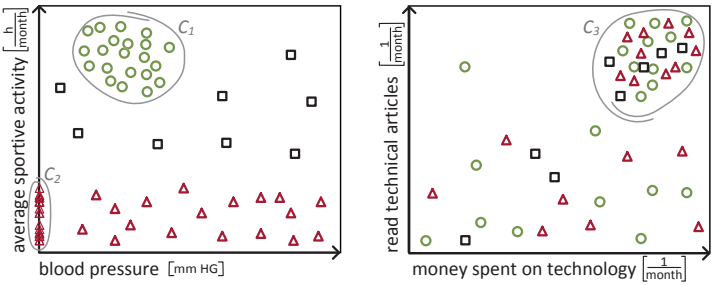


Figure 1.3: Exemplary subspace clusters in three different subspaces

space, where the influence of irrelevant attributes is weakened. Unfortunately, for complex datasets, different clusters may have different sets of relevant attributes, such that global dimensionality reduction techniques do not provide a satisfying solution. This problem of locally relevant dimensions for each cluster is explicitly tackled by the paradigm of subspace clustering [PHL04, KKZ09]. Subspace clustering is not restricted to a single data projection but detects clusters in arbitrary subspace projections of the data. For each cluster it automatically detects the set of relevant features for which the cluster's objects are similar. The relevant features of a cluster support a semantic reasoning about the data's clustering structure. The example in Fig. 1.3 shows a clustering consisting of three clusters each in a different subspace projection. Cluster C_1 is located in subspace {blood pressure, sportive activity}, while for the grouping of cluster C_2 only the attribute {sportive activity} is relevant and cluster C_3 is located in the disjoint subspace {money spent on technology, read technical articles}. Since different subspaces represent different characteristics of the data and, thus, might reveal clusters in a different semantic context, each object can naturally belong to multiple clusters simultaneously. Therefore, all three clusters of the example in Fig. 1.3 are potentially meaningful and should be reported as result.

1.3 Contributions and Structure of this Thesis

In this thesis, we want to present new models and algorithms for effectively combining the two paradigms of subspace clustering and multi-view clustering. Both paradigms share the fundamental belief that just a single partitioning of the data is often insufficient and that, instead, objects can be clustered differently depending on the context or view. While for multi-view or alternative clustering the term

"view" is not clearly defined and it is often hard to analyze the semantic behind discovered alternative clusterings, subspace clustering provides a natural intuition of a view on the data, as different subspace projections provide a different semantic perspective on the data. The two different 2-dimensional subspaces in

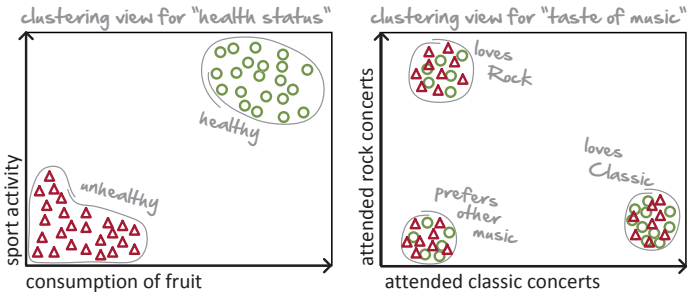


Figure 1.4: Different subspaces providing different clustering views on the data

Fig. 1.4 show two differing clusterings of a single dataset. Each set of characteristics represents a different view on the data, which enables to find such a diverse set of clusterings. Taking a perspective focused on health aspects, different individuals will be grouped together than when taking a perspective focused on musical or professional interests. Furthermore, the relevant attributes for each clustering support the identification of a semantic context for each clustering. A clustering based on the attributes "fruit consumption" and "sport activity" is likely related to the "health status" of a person, while the type of favored music concerts reveals a person's "taste of music".

A transfer of the principles of subspace clustering to multi-view clustering cannot only help in defining a view and a semantic background for alternative clusterings, it will also properly address the problem of irrelevant dimensions and account for the curse of dimensionality. Similarly, an adaption of the idea of clustering views and of the diversity of these views for subspace clustering can help in conquering one of the main challenges of subspace clustering, which is the avoidance of redundant clusters in the result set. Since the cluster definition of subspace clustering methods is often (nearly) anti-monotone w.r.t. the subspaces, each subspace projection of a valid cluster is a valid cluster as well. To avoid an overwhelming result set with redundant information, subspace clustering algorithms usually rely on a special redundancy modeling.

Part I: Introduction

In Chapter 1 of this first part of the thesis, we provided a short introduction for the two major data mining paradigms this thesis will cover: *multi-view clustering* and *subspace clustering*. In Chapter 2 we will continue with a discussion of the related work for both paradigms in order to capture the general approaches tackling both clustering problems. In addition to a formal problem formulation, we will also highlight the main challenges for each paradigm.

Part II: Transferring Multi-View Principles to the Subspace Clustering Paradigm

In the second part of this thesis, we take the perspective of subspace clustering and adapt an important principle of multi-view clustering to improve the clustering result. We start in Chapter 3 with a thorough discussion of the relation between multi-view clustering and subspace clustering to analyze similarities and differences.

In Chapter 4, we introduce the OSCLU approach which integrates the similarity of subspace perspectives into the model to handle redundancy in the result set. The redundancy of clusters is evaluated based on their similarity regarding objects and subspaces. The underlying idea is that only almost orthogonal subspaces are able to provide diverse clusterings. The OSCLU approach presents a general and flexible solution for detecting subspace clusters in different views of the data without relying on a specific cluster definition. Due to the NP-hard complexity of finding the globally optimal final solution, we propose an efficient algorithm to compute an approximate solution for the density-based cluster model.

The OSCLU approach finds a globally optimal set of alternative clusters regarding interestingness and redundancy. However, it cannot take prior information into account to determine a subspace clustering that represents a good alternative for already known subspace cluster information. In Chapter 5, we will present our ASCLU approach, which is a natural extension of the OSCLU model incorporating the information of a previously known subspace clustering into the global optimization process. The new clustering produced by ASCLU exhibits no redundancy to the apriori given clustering but complements its information to an overall optimal solution.

Part III: Transferring Subspace Principles to the Multi-View Clustering Paradigm

In the third part of this thesis, we take the perspective of multi-view clustering and transfer the idea of searching for clusters in subspace projections to the task of finding alternative clusterings. The two approaches OSCLU and ASCLU, presented in the previous part, focus on the clustering as a whole but views are only considered implicitly and are not mined explicitly. The views do not manifest themselves by assigning clusters to views and by determining which attributes are characteristic for which view. In Chapter 6, we will introduce the general idea for determining multiple alternative subspace clusterings simultaneously, to overcome these limitations. With generative models, we assume the data to be the result of a generative process depending on different mixture distributions for different subspaces, representing the clustering views.

In Chapter 7, we introduce our MVGen method which couples the detection of subspace clusters and their aggregating views. The generative model of MVGen considers classical subspace clustering scenarios, where a cluster has locally irrelevant dimensions for the view it is assigned to and where global noise dimensions can occur. To determine the relevant dimensions for each view and their subspace clusters, we perform Bayesian model selection, where we allow for overlapping subspaces for each view. Since learning the model variables through exact inference is intractable, we approximate the optimal solution using the principle of iterated conditional modes.

The generative model SMVC introduced in Chapter 8, is motivated by the success of the MVGen model. The subspace clustering scenario modeled with our SMVC model is simplified and the focus is directed towards a meaningful integration of user defined partial prior information regarding the clustering structure in the multi-view scenario. Via instance level must-link and cannot-link constraints the user is enabled to guide the complex clustering process towards a more satisfying result. Besides the difficulty of learning the clustering and the relevant subspaces for each view, a new task is to learn the association of the provided instance level constraints to the views. For efficient learning of the model variables, we use variational inference and mean field approximation techniques to approximate the optimal solution.

Part IV: Constraint-Based Alternative Clustering in Subspace Projections

Part III presented techniques to simultaneously find multiple clustering views hidden in the data. Most techniques for the multi-view clustering paradigm, especially if incorporating data transformations, instead, search for clustering alternatives iteratively. This has the advantage that based on the knowledge of previously found clusterings, the search for a new clustering is not completely uninformed but can be steered towards promising directions. In this part, we want to introduce a new concept for finding alternative clusterings in subspace projections based on methods for combined graph clustering of graph data and attribute data. Chapter 9 gives an introduction to the problem reformulation and thoroughly discusses the main challenges for this task. By encoding known clusterings as relational information between objects, the vector data and the known clusterings can be represented together as either vertex labeled graph or as edge labeled graph. In the chapters 10 and 11, we will present two techniques for performing subspace clustering in graphs annotated with feature vectors.

In Chapter 10, we propose the novel clustering method SSCG for graphs with vertex labels based on the principle of spectral clustering. Following the idea of subspace clustering, our method detects for each cluster an individual set of relevant features. Since spectral clustering is based on the eigendecomposition of the affinity matrix, which strongly depends on the choice of features, our method simultaneously learns the grouping of vertices and the affinity matrix.

In Chapter 11, we present the novel clustering method SuMo for graphs with edge labels. We extend the widely used modularity measure, used to express the strength of communities, for multi-dimensional edge weights by following the principles of subspace clustering. Some of the existing algorithms for approximating the optimal solution with respect to the traditional modularity can already be adapted for our extension of the modularity. To deal more effectively with the extended search space due to the variance of the dimensions relevance, we propose the efficient clustering algorithm SuMo for clustering networks based on the subspace modularity.

Part V: Evaluation and Visualization for Alternative Subspace Clustering

In this part of the thesis we will discuss measures and techniques for evaluating and visualizing clustering algorithms in the context of multiple views and subspace projections. We will consider post-processing techniques as well as in-process techniques for supporting the user in confining the clustering result.

In Chapter 13, we discuss the possibilities for a systematic evaluation of subspace clustering results. We formalize general quality criteria for subspace clustering measures and compare the existing external evaluation methods based on these criteria and pinpoint limitations. We propose a novel external evaluation measure which meets the requirements of the proposed quality properties. Overall, we provide a set of evaluation measures that fulfill the general quality criteria as recommendation for future evaluations.

Decoupling the process of subspace search from the actual clustering process provides more flexibility for the task of subspace clustering, e.g., regarding the choice of the cluster model. The choice of interesting subspaces is, however, crucial and for the choice of a proper clustering paradigm the user needs some analytical foundation. In Chapter 14, we propose an interestingness-guided subspace search method for facilitating the choice of subspaces by using the principles of different views. We provide visualization and navigation possibilities to interactively explore large sets of subspaces. Our approach allows users to effectively compare and relate subspaces with respect to involved dimensions and clusters of objects and facilitates the choice of appropriate clustering paradigms for selected subspaces.

In Chapter 15, we present two tools that help in bridging the gap between subspace clustering and multi-view clustering. Although subspace clustering methods generate concept-based patterns, the user has to provide domain knowledge to gain reasonable concepts or views out of the data. The first tool *CoDA* supports the user in the final step of view definition. More concretely, the user is guided through an iterative, interactive process in which views are suggested, analyzed, and potentially refined. Based on the views defined with *CoDA* or for the several alternative clustering solutions generated by multi-view approaches, our second tool, *MCExplorer*, allows for an interactive exploration, browsing, and visualization of multiple clustering solutions on several granularities.

Part VI: Summary and Outlook

In the last part, we conclude this thesis by summarizing all contributions and by presenting interesting open challenges in the context of multi-view clustering in subspace projections.

2
Related Work

THIS chapter will provide a rough overview of published clustering methods that are related to the task of finding multiple alternative clustering solutions in subspace projections. We will mainly focus on the two paradigms multi-view clustering (Section 2.1) and subspace clustering (Section 2.2).

2.1 Multi-View Clustering

The research field of multi-view clustering, which is also commonly referred to as alternative clustering, comprises various approaches applying different techniques and different assumptions concerning the data. Before discussing the different approaches, we start by formalizing the general mining task for finding multiple alternative clusterings.

Problem Definition 2.1 *Generalized Multi-View Clustering Problem*
Given a set of objects $O = \{o_1, \ldots, o_n\}$ *and a set of* $m \geq 0$ *known clusterings* $Known = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$ *as background knowledge, generate* $l \geq 1$ *alternative clusterings* $Alt = \{\mathcal{C}_1, \ldots \mathcal{C}_k\}$ *such that*

1. *the quality of the generated clusterings* $\sum_{\mathcal{C} \in Alt} Qual(\mathcal{C})$ *is maximized and*
2. *the similarity of all clusterings* $\sum_{\mathcal{C}_i, \mathcal{C}_j \in Known \cup Alt \wedge \mathcal{C}_i \neq \mathcal{C}_j} Sim(\mathcal{C}_i, \mathcal{C}_j)$ *is minimized.*

Like the solutions we will present in the next chapters, most approaches are developed for numerical vector data $O \subseteq \mathbb{R}^{|Dim|}$ with Dim being a set of dimensions. However, some approaches present general frameworks for discovering alternative clusterings and are independent of a specific cluster model such that they are not restricted to one data domain. Usually a clustering, for the set $Known$ as well as the set Alt, is considered to be a hard clustering and presents a partitioning of the data O.

2.1.1 Naive Approach

An intuitive approach to achieve a multi-view clustering is to randomly generate a variety of clustering solutions of which then a set of diverse clusterings is extracted in a post-processing step. The set of base clusterings can either be generated by using different clustering approaches, by using different parameter settings, or by exploiting the non-determinism or the trap of local minima of certain optimization algorithms. The post-processing step of selecting informative alternative clusterings will be a mining task itself. In [CENS06], the selection of clustering alternatives is solved through a meta clustering approach, where similar clusterings are grouped based on a similarity metric for clusterings. Representatives for the clusters at the meta level present the desired alternative clustering solutions. Besides of the inefficiency of the generation step, it also carries the risk of generating highly similar clusterings as well as clusterings of bad quality. Instead of such an undirected and independent generation of solutions, more systematic approaches would promise clusterings of higher quality and diversity.

Following the true meaning of multi-view clustering, we can find multiple, differing categorizations of the approaches for this paradigm. In the following, we will focus on just one categorization and differentiate between two processing schemes to systematically generate alternative clusterings. The first approach uses the knowledge of previous clusterings to iteratively generate new clustering alternatives. The second approach produces multiple alternatives simultaneously such that each clustering influences the others and a diversity can be realized.

2.1.2 Iterative Approach

Approaches that work iteratively assume a set of $m \geq 0$ known clusterings $Known = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$ based on which a single alternative clustering $\mathcal{C}_{alt}$ is generated ($l = 1$, cf. Problem Definition 2.1). Regarding the new clustering $\mathcal{C}_{alt}$ as known information $Known \cup \{\mathcal{C}_{alt}\}$, a further clustering alternative can be produced in a subsequent iteration, and so on. Most of the existing approaches have been designed to incorporate just a single known clustering ($0 \leq m \leq 1$, cf. Problem Definition 2.1). These approaches carry the risk that previously discovered clusterings are revisited in later iterations since the dissimilarity of new clusterings is just ensured for the one clustering of the previous iteration but not for *all* known clusterings. For some approaches a naive extension to incorporate $m > 1$ clusterings is easily possible, as we will discuss in the following.

The first approaches presented for this category find alternative clusterings based on the information bottleneck principle. The general idea of the information bottleneck principle is to find the best trade-off between accuracy and complexity (compression) when clustering data objects X (random variable representing object IDs) w.r.t. their attribute values Y (relevant variable representing the objects' attribute values). A (probabilistic) clustering C (random variable representing a clustering) should compress the information of X as much as possible (minimize mutual information $I(X, C)$) while preserving the information of the features Y (maximize the mutual information $I(Y, C)$). Overall, this trade-off can be realized via $min_{p(c|x)}\left[I\left(X, C\right) - \beta \cdot I\left(Y, C\right)\right]$, where β is the Lagrange multiplier realizing the trade-off. This variational problem formulation can be solved via a generalization of the Blahut-Arimoto algorithm [TPB99].

The information bottleneck objective, as defined above, represents the quality criterion for Definition 2.1. To incorporate the dissimilarity constraint for a given clustering D this objective has to be extended. Here, different ideas for using the information bottleneck with side information, i.e., a known clustering D, exist in the literature [CT02, GH03, GH04]. In [CT02] the main idea is to minimize the similarity of D and C by minimizing the mutual information $I(S, C)$, which is integrated as a third trade-off component into the objective function: $min_{p(c|x)}\left[I\left(X, C\right) - \beta \cdot I\left(Y, C\right) + \gamma \cdot I(D, C)\right]$. To avoid a third trade-off parameter, in [GH03] the redundancy of C and D is considered within the conditional mutual information $I(Y, C\,|D)$ of variables Y and C when already knowing clustering D. Maximizing this term ensures that the new clustering C provides novel knowledge: $min_{p(c|x)}\left[I\left(X, C\right) - \beta \cdot I(Y, C\,|D)\right]$. The authors introduce a relaxation in [GH04] and focus mainly on the diversity of the two clusterings while enforcing only a minimal quality threshold: $max_{p(c|x)}\left[I(Y, C\,|D)\right]$ such that $I(X, C) \leq c$ and $I(Y, C) \geq d$. Other similar techniques exploiting information theoretic principles can be found in the literature, e.g., [DB10a] which optimizes the objective $max_c\left[I(C, Y) - \beta \cdot I(C, D)\right]$.

The information theoretic approaches presented so far are designed to incorporate just a single known clustering. Here, for a new clustering the dissimilarity is just guaranteed for this single other clustering. A straightforward solution for considering multiple clusterings as prior information is to replace the occurrence of the redundancy term (e.g., $I(D, C)$) in the objective function with the sum of dissimilarities (e.g., $\sum_{D \in Known} I(D, C)$). Approaches following this idea are for example [GVG05, VE10, DB13a]. For binary data, [GVG05] presents a

method for likelihood maximization with model-level constraints to encode the known clustering information. [VE10] optimizes the same objective as [DB10a] based on conditional entropy and kernel density estimation. [DB13a] presents an elegant way to combine the minimization of the summarized mutual entropies $\sum_{D \in Known} I(D, C)$ and the likelihood maximization of the variables for the density mixture model: $\max_{\Theta} \left[L(\Theta \,|X) - \beta \cdot \sum_{D \in Known} I(D, C) \right]$, where $L(\Theta \,|X)$ is the log-likelihood function for determining optimal distribution parameters Θ.

All approaches presented so far use information theoretic principles for the quality and dissimilarity constraint of Definition 2.1 [CT02, GH03, GH04, DB10a, VE10, DB13a] or just to model the dissimilarity requirement [GVG05, DB13a]. Aside from information theory, various other techniques for ensuring the dissimilarity of the generated clustering have been published [GH05, BB06, BBD10].

[GH05] uses a heuristic approach that is based on ensemble clustering methods and can only incorporate a single known clustering $\mathcal{C}_{Known}$. The presented CondEns algorithm operates in three stages. First, the objects O_i of each known cluster $C_i \in \mathcal{C}_{Known}$ are clustered separately with an arbitrary traditional clustering method. Since for each of the $k = |\mathcal{C}_{Known}|$ many newly generated clusterings $\{\mathcal{C}_{C_1}, \ldots, \mathcal{C}_{C_k}\}$ only the objects of one cluster C_i have been considered, the remaining objects $O \setminus O_i$ will be properly assigned to the clusters for each clustering $\mathcal{C}_{C_i}$. In a final step, a single alternative clustering is generated out of these k base clusterings by using ensemble clustering techniques. A crucial assumption for the success of CondEns is that each of the known clusters $C_i \in \mathcal{C}_{Known}$ has to contain information about many or even all of the alternative clusters.

The Coala algorithm, presented in [BB06], encodes a known clustering with instance-level cannot-link constraints and uses an agglomerative hierarchical clustering to realize a trade-off between quality and dissimilarity. For each object pair, it introduces a cannot-link constraint if both objects appear together in one of the known clusters, indicating that those two objects should not be grouped together again in order to achieve a novel clustering structure. For the merging steps of the hierarchical clustering, a trade-off is realized between quality merges d_{qual} and dissimilarity merges d_{diss}, i.e., d_{qual} is the distance of the two closest clusters and d_{diss} is the distance of the two closest clusters such that no cannot-link constraints are violated. Only if the quality merge is significantly better ($d_{qual} < w \cdot d_{diss}$) according to the trade-off parameter w, it will be preferred over the dissimilarity merge. Although this principle can be easily adapted to multiple known clusterings algorithmically, it becomes very likely that already

for few known clusterings the quality merge will always dominate and thus a single clustering will constantly be reproduced.

For the MAXIMUS approach in [BBD10], Bae et al. even develop the new similarity measure ADCO for clusterings that emphasizes structural dissimilarity (w.r.t. the clusters' density profile) and can deal with non-overlapping clusterings. Used as an objective, the optimization problem for minimizing ADCO w.r.t. multiple known clusterings can be encoded as an integer linear program, whose localized clustering solutions are combined with a consensus clustering process.

All approaches we presented so far for the category of iterative multi-view clustering, cluster in just a single data space and explicitly consider the dissimilarity of the generated clustering solutions as part of their objective. The algorithms in [CFD07, DQ08, QD09, DB13b] follow a different approach, where different data representations are considered for each clustering. The general principle is to learn an "orthogonal" transformation of the data based on a previous clustering result. The idea is that the new data representation can highlight novel clustering structures, which is strongly related to the subspace clustering paradigm. These techniques do not explicitly check for the dissimilarity of the generated clusterings but only implicitly account for the diversity through differing space transformations. The general aim for these approaches is to find a transformation of the data that is independent of the known clustering but at the same time preserves quality characteristics of the data to avoid its complete distortion. Therefore, they are usually restricted to linear space transformations. A big advantage of the transformation-based approaches is their independence of a specific clustering model. For each of the determined data transformations an arbitrary (preferably partitioning) clustering model can be applied.

The oldest approach based on data transformations [CFD07] exploits dimensionality reduction techniques. For a given clustering the main factors (principal components) leading to this clustering are identified. By removing these main factors characterizing the previous clustering, only the residual, orthogonal space is considered for the next clustering. Thereby, previously weak principal components are highlighted, which can support alternative clustering structures. By iteratively generating new space transformations based on the previously generated ones, this approach incorporates all previously known clusterings. However, the repeated projection of the data into reduced spaces can quickly merge the data into a single cluster. Furthermore, this approach might not be appropriate for lower-dimensional datasets [DQ08].

The approach of [DQ08] uses instance level constraints to characterize an existing clustering $\mathcal{C}_{Known}$. Based on a metric learning algorithm for these constraints, a transformation $T_{\mathcal{C}_{Known}}$ is determined such that the known clustering is easily observable. Via a singular value decomposition $T_{\mathcal{C}_{Known}} = L \cdot A \cdot R$ an "inverse" transformation $T'_{\mathcal{C}_{Known}} = L \cdot A^{-1} \cdot R$ is determined which rules out the previously found clustering but maintains the inherent structure of the data. Although, this procedure can be applied iteratively like [CFD07], there is no guarantee for the dissimilarity of the transformations and correspondingly the dissimilarity of the resulting clusterings.

The approach of [QD09] solves a constrained optimization problem to find a good transformation of the data based on a single known clustering. It minimizes the Kullback-Leibler divergence between the distribution of the original data and the one of the transformed data without overly distorting the data properties. For the optimization process, the authors also propose a trade-off possibility such that the user is able to favor either alternativeness or quality. This approach additionally offers the nice opportunity to specify certain parts of the known clustering which should be retained. Similar to [DQ08], a naive extension for multiple known clusterings does not guarantee a new data transformation which is dissimilar to previous ones.

In [DB13b], a globally optimal subspace is learned using regularized PCA such that the new subspace is independent from a given clustering and at the same time naturally preserves the characteristics of the data. To achieve the independence of subspaces, the authors employ the Hilbert Schmidt Independence Criterion (HSIC), which, in combination with PCA, can lead to an eigendecomposition problem for which a globally optimal solution can be derived. For more complex data structures, an alternative way to compute a subspace projection based on graph theory is proposed, which aims to preserve the neighborhood proximity of the data objects. Again, a naive extension for multiple known clusterings does not guarantee alternative subspace projections.

2.1.3 Simultaneous Approach

While the previous paradigm iteratively searches for alternative clusterings, approaches of the simultaneous paradigm try to determine all clustering solutions in parallel. This has the advantage that all clustering solutions influence each other such that the overall quality of the generated alternatives $\sum_{\mathcal{C} \in Alt} Qual(\mathcal{C})$

can be maximized. Approaches with an iterative clustering detection scheme, instead, perform a greedy selection of the best available clusterings. Only subsequent clustering solutions are to be adapted to guarantee a diverse set of clusterings but already generated clusterings cannot be modified. An affiliated effect is that mistakes in previous iterations leading to bad clusterings can negatively influence subsequent clusterings. Approaches simultaneously detecting multiple clusterings typically do not incorporate any given clustering ($m = 0$, cf. Problem Definition 2.1) and the number of generated alternatives ($l \geq 1$) is usually defined by the user.

In [JMD08] two approaches are proposed which use the notion of decorrelation between clusterings, which quantifies the "orthogonality" between the mean vectors corresponding to different clusterings. The first approach modifies the k-means algorithm to find compact clusterings, where the representatives of different clusterings should be mostly orthogonal to each other such that the cluster labels generated by nearest-neighbor assignments are independent. The second approach presents a generalized expectation maximization algorithm for learning the convolution of multiple independent mixture distributions.

The CAMI approach [DB10b] exploits a regularized expectation maximization technique, which maximizes the likelihood of each alternative clustering over the data and simultaneously minimizes the similarity between them based on the mutual information. As for the approach in [JMD08], the different clusterings are learned as a convolution of multiple independent mixture distributions.

The work of [HTW+10] focuses on non-homogeneous data, where two different object domains are considered, whose instances can additionally be connected through relational information (bipartite graph). The task is to find a base clustering for the objects of each data domain such that either the according contingency table shows a strong diagonal (dependent clustering, relations between the two clusterings are strong) or such that the according contingency table is uniformly distributed (disparate clustering, relations between the two clusterings are weak). The minimization or maximization of an integrated objective function leads to a disparate or a dependent clustering. Although this approach can only generate two alternative clusterings, it provides a very general framework by considering two different databases and relational information.

The information theoretic model presented in [KdB13] follows the principle that clusters are more interesting if their probability is small under some prior beliefs. The prior beliefs can be simple distributional assumptions regarding the

data but can also include already known clusters. The probability of clusters or a set of clusters is derived based on a maximum entropy model of prior beliefs. However, optimizing a set of alternative clusters is NP-hard, such that the authors propose a greedy approximation algorithm where clusters are generated iteratively and integrated into the prior belief for subsequent cluster generations. Although this shall approximate the simultaneous computation of multiple, independent clusters, we can also argue to assign this approach to the iteratively operating multi-view clustering paradigm. It is also important to note that this approach does not produce a set of clusterings, but a set of non-disjoint clusters.

So far, [NDJ10] is the only approach aiming at simultaneously learning all clustering views based on multiple data representations. By augmenting the objective for spectral clustering views to incorporate multiple views, dimensionality reduction, and a penalization for similarity of the views, multiple clustering solutions in subspace projections of the data are determined.

2.2 Subspace Clustering

Since the first subspace clustering approach [AGGR98] has been published in 1998 , numerous new methods have been proposed for which roughly two different paradigms can be distinguished: subspace clustering [AGGR98] and projected clustering [AWY+99]. Both paradigms tackle the general problem of finding clusters in subspace projections of the attribute space. The most general problem formulation for subspace clustering also includes non-axis-parallel subspaces and can be formalized as follows:

Problem Definition 2.2 *Generalized Subspace Clustering Problem*
Given a set of objects $O = \{o_1, \ldots, o_n\} \subseteq \mathbb{R}^{|Dim|}$ *with* Dim *being a set of dimensions* $Dim = \{1, \ldots, d\}$*, find a subspace clustering* $\mathcal{C} \subseteq 2^O$*, such that for each cluster* $C_i = (O_i) \in \mathcal{C}$ *a linearly transformed space* $S_i = f_{O_i}(Dim)$ *exists such that* C_i *is of high quality in this projected space* S_i*.*

The linear space transformation can for example be the result of a PCA transformation. For these arbitrary subspace projections the search space of potential clusters becomes infinitely large, such that heuristics are applied to confine the search. In this thesis the focus, however, will be on approaches for clustering in axis-parallel subspaces for which a cluster can be understood as a pair consisting of the set of its clustered objects and the set of its relevant attributes.

Problem Definition 2.3 *Generalized Axis-Parallel Subspace Clustering Problem*
Given a set of objects $O = \{o_1, \ldots, o_n\} \subseteq \mathbb{R}^{|Dim|}$ *with* Dim *being a set of dimensions* $Dim = \{1, \ldots, d\}$*, find a subspace clustering* $\mathcal{C} \subseteq 2^O \times 2^{Dim}$*, such that each cluster* $C_i = (O_i, S_i) \in \mathcal{C}$ *has a high quality in its respective subspace* S_i*.*

While there is no generally accepted definition for the quality of a clustering, all existing measures somehow consider the proximity or similarity of the clustered objects.

2.2.1 Subspace Clustering

Subspace clustering approaches base on a formal definition of what constitutes a cluster based on the similarity of an object set for a subset of attributes. Among the various presented approaches different definitions for subspace clusters exist [PHL04, KKZ09]. The general idea, as introduced by [AGGR98], is to find all pairs of object sets and subspaces that match this definition. The object sets usually have to fulfill a maximality criterion, i.e., for a given pair (O, S) there should not exist a superset $O' \supseteq O$ such that (O', S) also is a valid cluster.

The first challenge of subspace clustering clearly is the computational complexity. Given the exponential number of potentially interesting subspaces, most subspace clustering algorithms have very high runtimes. Often the problem is slightly diminished by exploiting heuristics to approximate the result.

An important characteristic of subspace clustering results is that the clusters are allowed to overlap with respect to objects as well as attributes. On the one hand, this allows that each object can participate in multiple clusters and, thus, enables us to find multiple concepts hidden in the data. On the other hand, one has to cope with the potentially tremendous amount of all possible clusters in the exponential number of axis-parallel subspaces. Typically, the cluster definition fulfills an anti-monotonicity criterion such that a set of objects forming a valid cluster in subspace S also represents a valid cluster in all subspaces $S' \subseteq S$. Therefore, the result size of subspace clustering algorithms can be huge and quickly become unmanageable [MGAS09]. Usually, the projected versions of a cluster in its subspaces do only comprise little novel objects and, therefore, can be regarded as redundant information. The second challenge of subspace clustering, thus, is the redundancy of resulting clusters. Some approaches have been proposed that explicitly incorporate a redundancy pruning of the result [AKMS07a, AKMS08a, AKMS08b, MAG+09b, MS08]. In addition to the cluster definition, they often define an optimal clustering to report only the most

interesting clusters, which provide novel knowledge about the data. In addition to the exhaustive search of the exponentially many subspaces, thus, typically a complex optimization task has to be solved.

A third challenge for subspace clustering is to adequately consider the effects of the curse of dimensionality. By considering subspace projections, the negative influence of irrelevant dimensions can be avoided. The decreasing density of objects with increasing dimensionality, however, also applies to clustered objects in subspaces. Using just a single cluster definition for all subspaces, independent of the subspace cardinality, might prevent the detection of meaningful clusters. Some approaches like [SZ04, AKMS07a, MS08] explicitely account for the problem of a decreasing density and use a dimensionality unbiased subspace cluster definition, e.g., by involving statistical significance thresholds [SZ04, MS08] or by normalizing the density w.r.t. the null model [AKMS07a].

Most of the approaches presented for subspace clustering evolved from traditional clustering models like grid-based approaches [JD88], DBSCAN [EKSX96], K-Means [Mac67], or EM-based techniques [MK08]. Besides these approaches aiming for clusters that excel by a high compactness or density, some approaches aim for object groupings that describe correlations of different attributes, so called correlation clustering methods (e.g., [AY00a, BKKZ04, ABK+07b, ABK+07a, AR10, ABD+08, HH07]). Since the clusters' dimensions are not restricted to subsets of the original attributes but correspond to arbitrarily oriented subspaces, correlation clustering is often denoted as generalized subspace clustering. This, however, is inaccurate because of two reasons: First, existing correlation clustering methods are not able to find multiple overlapping clusters, since they are limited to find only disjoint or, in the case of [AR10], nearly disjoint clusters. Even more serious is the ignorance of the obfuscation provoked by highly overlapping clusters in different subspaces, causing most approaches to fail in detecting the true correlation clusters. With our SSCC approach presented in [GFVS12], we adequately transfer the principles of subspace clustering to the problem of finding correlation clusters by analyzing subspace projections to find correlated dimensions supported by a subset of objects.

2.2.2 Projected Clustering

The paradigm of projected clustering, introduced by [AWY+99], aims at partitioning the data into disjoint clusters such that with each group its relevant dimensions are discovered simultaneously. Each object is assigned to exactly one

cluster, which is the major difference to the subspace clustering paradigm discussed before. Focusing on a partitioning of the data addresses two of the three main challenges for subspace clustering. First, the search space is decreased by considering a partitioning, which results in a better efficiency of approaches of this paradigm. Second, limiting the result to only disjoint clusters can be regarded as maximal redundancy elimination, such that the result size is manageable. However, by enforcing disjoint clusters, several meaningful clusters are only detected incompletely or are even lost entirely. In general, projected clustering methods are not able to detect multiple clustering views per object.

2.3 Bridging the Gap

Summarizing the related work for multi-view clustering (cf. Table 2.1), we have seen that most of the presented approaches focus on the iterative processing scheme for clustering in just a single data space. Comparably, few approaches deal with space transformations or a simultaneous detection of multiple clustering alternatives. Here, we will present new approaches in this thesis.

		iterative	**simultaneous**
single space:	2 alternatives	✓	✓
	$\geq$ 2 alternatives	✓	✓
transformed spaces:	2 alternatives	✓	✗
	$\geq$ 2 alternatives	(✓) just [CFD07]	(✓) just [NDJ10]

Table 2.1: Overview over the related work for multi-view clustering approaches

Searching for multiple clustering alternatives in just a single data space is not very promising. Since only a single data representation is considered, the detected clusterings, which depend on this single data distribution, will not differ to a high extent. Approaches working with a single data representation are forced to trade-off quality and diversity of the detected clusterings. A more promising approach is to use different data representations, where novel structures of the data might be hidden. Although agreeing that strong data distortions should be avoided to guarantee a meaningful clustering, approaches presented for this category use arbitrary linear transformations, such that results are often difficult to interpret. We argue that alternative clusterings can be expected especially for high-dimensional data, where different explanations of the data can be discovered through different characteristics of the data, i.e., distinct subspaces of the data. Within this thesis we will present approaches for detecting multiple alternative clusterings simultaneously (Part II & III) as well as iteratively (Part IV).

Part II

Transfering Multi-View Principles to the Subspace Clustering Paradigm

All truths are easy to understand once they are discovered; the point is to discover them.
GALILEO GALILEI

3

The Relation of Multi-View Clustering and Subspace Clustering

BESIDES the similarities of subspace clustering and multi-view clustering, we can also identify inherent differences which clearly distinguish both paradigms from each other. They both agree that for most data just a single partitioning of the data is not sufficient but that different perspectives on the data can reveal multiple, differing clusters for the same objects. A large customer database, for example, allows for different groupings depending on whether personal or professional preferences form the basis of clustering. Multi-view clustering algorithms usually do not focus on defining or finding a meaningful perspective on the data to reveal a new clustering. Instead they try to enforce, e.g., by using constraints, to find highly differing clusterings. Subspace clustering, on the contrary, offers a nice intuition of a perspective on the data. The belief here is that different sets of attributes highlight different characteristics of the data which, as a consequence, can lead to different clustering solutions. This not only provides a possibility for searching for new clustering alternatives but also enables a semantic interpretation of the resulting clusters that is often not possible for multi-view approaches. In Fig. 3.1 the attributes “average fruit consumption” and “sport activity” provide a different clustering perspective than the attributes “attendance to rock concerts” and “attendance to classic concerts”. We observe for each customer multiple possible behaviors which should be detected as clusters. Each behavior of a customer is described by specific attributes. Thus, meaningful clusters appear only in these specific subspace projections of the data. While the attribute “attendance to rock concerts” is useful for the distinction of musical interests, the attribute “fruit consumption” is irrelevant for grouping musical interests of customers. Furthermore, the relevant attributes for each clustering strongly support the semantic reasoning about found clusters and their views. Thus, we might label the customers showing high values for fruit consumption and sport activity

as “healthy” and those with low values as “unhealthy”. The respective clustering perspectives can thus be labeled as, e.g., “health status” and “taste of music”.

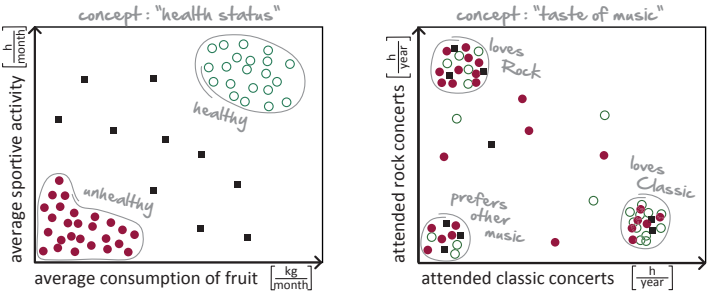


Figure 3.1: Different subspace projections reveal different views on the data

While approaches for multi-view clustering always search for multiple partitionings of the data, subspace clustering algorithms do not enforce partitionings but identify clustered regions in subspace projections of the data. This way, we believe subspace clustering to be better guided by the data itself than partitioning approaches. In this chapter, we want to take the perspective of subspace clustering and explore the possibility to integrate the underlying goal of multi-view clustering. Thereby, we want to preserve the beneficial characteristics of subspace clustering, namely:

- every cluster might have its individual set of relevant dimensions
- objects might belong to multiple clusters
- not every object needs to be clustered, i.e., there might exist outliers
- the final result clusters are determined simultaneously

Although both paradigms aim at revealing the multifaceted nature of the data by allowing objects to be clustered multiple times, they both also try to limit the result to a manageable size. Multi-view clustering limits the result set to only those clusterings that are of high quality and that are dissimilar to each other. For subspace clustering a cluster definition implicitly comprises a certain quality demand. The actual pruning of the result set is realized by filtering out redundant clusters. The underlying goal is, thus, the same for both paradigms and, put in a nutshell, the objective is to maximize the gained information while keeping the result size as small as possible.

What is called dissimilarity for multi-view clustering is called avoidance of redundancy for subspace clustering. While both intend to restrict the result set to only informative clusters, we can observe that the term dissimilarity has a broader conception than redundancy. While redundancy-avoidance is commonly considered to retain complete information and only to resolve duplicate information, dissimilarity is a very subjective term and, as we will see, has various interpretations in the literature. In the following, we will provide a brief overview over techniques to constrain the result size of both, subspace clustering and multi-view clustering.

3.1 Redundancy Avoidance for Subspace Clustering

Subspace clustering automatically detects clusters in arbitrary subspace projections. These clusters might overlap object and dimension-wise, i.e., objects can be part of various clusters in different subspaces. As a consequence, subspace clustering techniques have to cope with an exponential number of subspace clusters. Many of these clusters detect more or less the same groups of objects in similar projections of the data and, thus, provide no additional information. Given the typically huge result size of subspace clustering algorithms (cf. [MGAS09]), which might even exceed the number of objects to be clustered, the obfuscation of the actually interesting cluster information by redundant clusters becomes a severe problem. Therefore some approaches have been proposed that explicitly address this redundancy problem. Besides a definition of what constitutes a subspace cluster, they formulate a redundancy definition to confine the set of all possible subspace clusters to only the most interesting ones. We can divide these redundancy models into those with local and global scope.

Representatives of the first category [AKMS07a, AKMS08a, AKMS08b] base the redundancy definition on local cluster properties. Clusters are compared pairwise, such that a subspace cluster is redundant if it shares a user-specified fraction of objects with another cluster. Among the redundant clusters those with maximal information, i.e., the ones with more relevant dimensions, are chosen for the final result set. These maximal subspace clusters tend to contain less noise and thus represent the inherent clustering structure more accurately. The restriction to only pairwise comparisons of clusters fail to detect the redundancy of clusters that are covered by combinations of high dimensional subspace clusters. Reconsidering our toy example in Fig. 3.2, the additional benefit of knowing

cluster C_{10} is almost negligible if we already know clusters C_7 and C_8. A pairwise comparison of C_{10} to C_7 or to C_8, however, indicates a high fraction of newly clustered objects which is, in fact, not true.

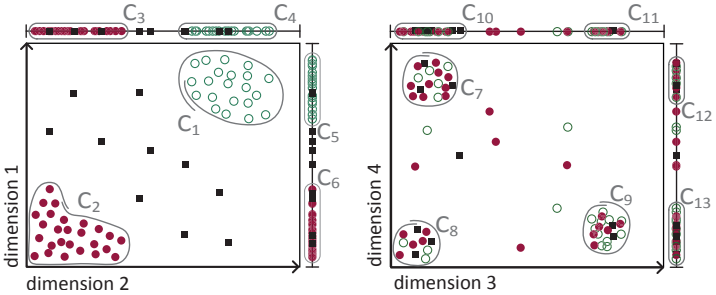


Figure 3.2: Redundancy introduced by subspace projections of clusters

Acknowledging this shortcoming of local redundancy models, global redundancy models [MAG+09b, MS08] compare the information of single clusters against the set of all known clusters. For the redundancy model of RESCU [MAG+09b], a cluster is non-redundant to a given set of clusters if it can contribute with a sufficient coverage of beforehand unclustered objects. While this definition solves the problem of detecting the redundancy of cluster C_{10} with respect to clusters $\{C_7, C_8\}$ in Fig. 3.2, it also easily expels the alternative clustering $\{C_7, C_8, C_9\}$ as redundant if the clustering $\{C_1, C_2\}$ is already known. The StatPC approach [MS08] defines a cluster as redundant if its support can already be approximately estimated with the information provided by the set of known clusters. The goal is to find all and only those clusters that are statistically interesting. While this approach allows to detect alternative subspace clusterings in general, it exhibits some flaws hindering its broad application. First, it is based on the assumption of uniform distribution inside a cluster. Second, the redundancy model is limited to the fixed cluster definition of StatPC.

3.2 Dissimilarity Criteria for Multi-view Clustering

The prevailing goal of multi-view and alternative clustering approaches is to find multiple clusterings that *highly differ* and are of *high quality*. Since the second requirement of a high quality holds for all clustering paradigms and various objective functions have already been presented, the clou of the multi-view clus-

tering paradigm clearly is the simultaneous realization of high quality and high diversity. Among the various presented solutions for the clustering problem of multiple views, we can identify two main categories regarding the technique for attaining disparate clusterings.

Approaches of the first category directly ensure the dissimilarity of clusterings via dual optimization techniques to simultaneously optimize the quality and the dissimilarity of the new clustering. The most popular approach is to integrate an according dissimilarity measure as constraint into the objective function. Here, mainly information theoretic measures have been applied [CT02, GH03, GH04, DB10a, VE10, DB13a, GVG05] but in [BBD10] Bae et al. also develop the new measure ADCO that emphasizes structural dissimilarity and can deal with non-overlapping clusterings. Besides the approaches directly involving a similarity measure for clusterings, in [BB06] Bae and Bailey use instance-level constraints and integrate their degree of violation into the objective function. The problem of such dual optimization approaches is the trade-off between quality and dissimilarity of the generated clusterings, which is usually realized by a user specified trade-off parameter. Enforcing a higher degree of dissimilarity is, thus, typically accompanied by a loss of the clustering quality. Since all the above approaches work in a fixed data space, there is no possibility to circumvent this trade-off.

For the second category, the approaches do not directly enforce the dissimilarity of the clusterings but use different perspectives on the data based on previous clusterings and certain assumptions in order to find novel clustering structures. The most common approach is to use a previous clustering to determine an orthogonal data space either through distance matrix transformations [DQ08, QD09] or through subspace projections [DB13b, CFD07]. A different technique is the one of [GH05], where Gondek and Hofmann propose a general framework that is based on data subsets and ensemble techniques. Approaches of this second category do not search just in the original data space but iteratively transform and cluster the data. The transformation of the data, which is learned based on the clustering structure of a previous result, is supposed to highlight novel structures and presents an opportunity to find a novel clustering of high quality. All approaches share the advantage that for the transformed data space any clustering method can be applied depending on the data properties. A common disadvantage of all approaches is, however, the solely implicit consideration of quality and dissimilarity of the new clustering based on certain assumptions. The orthogonal space transformations do not ensure that the new space reveals

a good clustering structure. Similarly also the dissimilarity of a new clustering to previous results is based on certain assumptions but cannot be guaranteed. This problem becomes most apparent with the two approaches of [CFD07] and [GH05]. In [CFD07] the data space is iteratively projected into non-paraxial subspaces based on the clustering of the previous iteration. Thereby, the data quickly collapses into a single cluster or even a single coordinate [DQ08] such that no meaningful cluster structure exists anymore. In [GH05] the underlying assumption is that, given a known clustering each new cluster label is present in each of the known clusters. If this assumption does not hold, the clustering quality as well as the dissimilarity of the new result is questionable. Furthermore, especially for the approaches [GH05, DQ08, QD09, CFD07], a global diversity of multiple clusterings is uncertain since only one clustering can be taken into account for the transformation.

3.3 Basic Idea for Combining Both Paradigms

While those approaches that solely cluster in the full-space are restricted with respect to the diversity of the clusterings and are bound to a trade-off between quality and diversity, those approaches that rely on space transformations share the philosophy of subspace clustering that new perspectives on the data can reveal new clustering structures. In the following Chapters 4 and 5, we want to present two approaches that try to transfer the general idea of finding alternative clusterings based on data transformations to the paradigm of subspace clustering. Thereby we want to inherit the advantages leading to the discovery of multiple alternative clustering concepts and help to tackle the redundancy problem for subspace clustering, whose benefits can in return help to overcome the conceptional problems of alternative clustering solutions.

The alternative clustering algorithms relying on space transformations assume that different space projections naturally contain different clusterings. They do not involve other techniques to guarantee the difference of the iteratively generated clusterings. While subspace clustering techniques look at the difference of clusters more carefully by comparing the set of covered objects, they do not incorporate the idea of the difference of space transformations. Either they compare clusters only against clusters in superspaces (local redundancy: [AKMS07a, AKMS08a, AKMS08b]), or they compare a cluster against the entire set of clusters (global redundancy: [MAG+09b]). While local redundancy cri-

teria are considered to be too weak and global criteria are often too restrictive, the additional consideration of the subspace similarity instead of only the object coverage can present a solution to this problem.

The quality of subspace clustering results is mainly determined by the structural quality of the detected clusters as well as the completeness and the redundancy of the set of clusters. The quality of the clusters is completely determined by the underlying subspace cluster definition. Given the evolution of subspace clustering methods, where the difficulty of redundancy has been discussed and explored only since the last decade, we can say that the quality of the clustering is the main focus of subspace clustering approaches. By remaining faithful to these principles of subspace clustering, we are not endangered of resulting in low quality results as multi-view approaches typically do.

In Chapter 4, we will present a new subspace clustering approach that simultaneously searches for all clusters of different clustering views. In Chapter 5, we will extend this model in order to incorporate an existing, known subspace clustering into the process to which an alternative subspace clustering needs to be found. The general idea of using the subspace similarity in order to avoid redundancy in the subspace clustering result will also reappear in Chapter 14, where we will present an interactive visual approach for decoupling the processes of subspace search and clustering.

4

Detection of Orthogonal Subspace Clustering Concepts

SINCE multi-view clustering and subspace clustering share the fundamental belief that objects might not belong only to a single cluster but can belong to multiple clusters simultaneously, it lends itself to combine both paradigms. In this chapter, we want to transfer the ideas of multi-view clustering to subspace clustering. We will see that the concept of clustering alternatives can help to tackle one of the main challenges of subspace clustering, namely to identify only the relevant clusters.

In this chapter, we propose the novel clustering method OSCLU (Orthogonal Subspace CLUstering) which aims at the detection of orthogonal concepts in subspace projections of the data. OSCLU reveals the clusters of (almost) orthogonal concepts described by different attribute subsets and prunes clusters of concepts that are too similar. Thereby, OSCLU is able to detect all interesting clusters in different views of the data while simultaneously avoiding redundant information.

4.1 Introduction

As thoroughly discussed in Chapter 3, one of the main challenges for subspace clustering, besides the efficiency, is the avoidance of redundant clusters in the result. Since most subspace cluster definitions fulfill at least a weak anti-monotonicity criterion, most approaches face the problem that a valuable cluster $C = (O, S)$ is also detected in most of its exponentially many subspaces $S' \subseteq S$ without further information value. The local ([AKMS07a, AKMS08a, AKMS08b]) or global ([MAG+09b, MS08]) redundancy criteria to tackle this problem proposed so far, are often considered to be too weak or too restrictive respectively. The most common approach is to compare the objects newly covered by a cluster compared to either all clusters or single clusters in superspaces. A new perspective for this problem is given by alternative clustering approaches that rely on data transformations [DQ08, QD09, DB13b, CFD07]. The underlying idea of these approaches is that a new clustering structure can be revealed if the data is transformed orthogonally to the information of a given, known clustering. This transformation does not ensure a good or different clustering structure compared to the known one, but it shares the observation of subspace clustering, that new perspectives (e.g., subspaces) can reveal new clustering structures, where the same objects can appear in different clusters than before.

The idea of the model presented in this chapter is to combine the two paradigms of subspace clustering and alternative clustering by integrating the difference or orthogonality of subspaces into the redundancy criterion, i.e., we allow clusters to overlap with respect to their objects if they are located in different subspaces. In general, we call these multiple *orthogonal concepts* that provide different *views* on the data. For the general case of high dimensional data, a concept is described by a subset of attributes and by the set of clusters located in this or a similar subspace. The concept's dimensions provide the semantic background why specific objects are grouped together. Focusing on axis-parallel subspaces makes such a semantic interpretation easier compared to the approaches of [DQ08, QD09, DB13b, CFD07]. In Fig. 3.1, some healthy and unhealthy customers group together in a 2-dimensional subspace and describe the concept "health status" and we can detect the orthogonal concept that represents the customer's "taste of music" in a 2-dimensional subspace that is orthogonal to the first one. Apart from our example of customer segmentation, similar observations can be made in the other scenarios as well: Genes are controlling multiple functions

(concepts) expressed only under specific conditions (relevant attributes for the concept), or sensors are measuring multiple concurrent environmental events (concepts) specified by different sensor measurements (relevant attributes).

A concept can contain several groups that are clearly separated in the relevant dimensions of the concept, like customers loving Rock or customers loving Classic in our previous example. While in similar subsets of relevant attributes clusters that have many objects in common introduce redundancy, for different sets of attributes objects can be clustered in multiple orthogonal concepts. Considering the concept "health status", a "Rock Fan" can be clustered with other customers to form a new grouping. There might exist multiple meaningful groups for each object as it can be interpreted in multiple different ways. Unlike the approaches for multi-view or alternative clustering, OSCLU, as a subspace clustering method, does not enforce each object to be clustered. As we can see in Fig. 3.1, within each concept there exist some outliers that do not belong to any of the concept's clusters. Our novel OSCLU (Orthogonal Subspace CLUstering) approach detects for each object multiple orthogonal concepts. Each detected cluster provides novel information, as we aim at detecting only clusters in orthogonal subspaces.

Summing up, in our approach, we aim at the detection of only the orthogonal concepts fulfilling the following properties:

- subspaces and subspace clusters represent the concepts in the database
- objects might be present in multiple clusters if the subspaces of their concepts differ (to a high extent)
- each cluster provides novel information for its concept

Following these properties, we propose a method for selecting orthogonal subspaces by using a similarity measure on subspace projections. According to this similarity our novel approach OSCLU chooses only the clusters in orthogonal subspaces for the result set. In addition, we propose a relaxation of the orthogonal subspaces to "almost orthogonal subspaces". This generalization allows us to detect concepts sharing a certain amount of common dimensions. The attribute "gender" for example could belong to several concepts. A relaxation to almost orthogonal subspaces admits more concepts to the result.

As each object might be present in multiple clusters, we have to ensure that each cluster adds sufficiently novel information within its concept. Unlike most subspace clustering techniques, we prevent redundant information. For this purpose, we introduce an interestingness measure for choosing only sufficiently distinct clusters from similar concepts. Furthermore, to select the most interesting

clusters, we present an objective function that is based on multiple properties like size, dimensionality, and density of the subspace clusters. Using both properties of orthogonal subspaces and most interesting clusters, OSCLU performs a global optimization of the result set. It ensures to include overlapping clusters to detect multiple concepts. Furthermore, it prunes similar subspaces and non-interesting clusters to ensure all patterns in the result are meaningful.

4.2 Orthogonal Concepts in Subspaces

In this section, we present our model for the detection of orthogonal concepts in subspaces of high dimensional data. Formally, we map our contributions to an optimization problem based on detected subspace clusters in the database. In contrast to subspace clustering, where all clusters are selected for the result set, we choose only a subset of most interesting clusters based on orthogonal subspaces. For this, we make a distinction between the cluster definition and the clustering definition. While the cluster model defines the properties that a set of objects $O \subseteq DB$ and a set of dimensions $S \subseteq Dim$ have to fulfill to be a valid cluster $C = (O, S)$, the clustering model determines a set of clusters $M = \{C_1, \ldots, C_n\}$ to be a valid clustering. The valid clustering for traditional subspace clustering is simply the set containing all subspace clusters (All). This set is highly redundant and hence in our model it is not a valid clustering.

We want to generate a highly informative clustering $Opt \subseteq All$ such that the clusters in the result set represent the multiple concepts of the data without obfuscating this structure by redundant information. As motivated before, each object might be present in multiple clusters if the clusters describe different concepts and each cluster $C \in Opt$ has to provide novel information within its similar concepts. In short, it is not allowed to group the same objects in similar concepts by several clusters. Therefore, we have to define

- if a concept is similar to another one or if it describes a different concept
- and if a cluster identifies a new grouping within its similar concepts.

As a consequence overlapping clusters between different concepts are possible, in contrast to projected clustering. We solely have to check if the same objects are already described within similar concepts to filter out uninteresting clusters and to steer our cluster detection to the orthogonal subspaces. Thus, in a first step, in Section 4.2.1, we define the notion of (almost) orthogonal concepts,

to determine which concepts are similar to a selected one. In Section 4.2.2, we present the interestingness criterion, that each cluster has to fulfill to be an informative cluster within its similar concepts. In Section 4.2.3, we define our overall model for the optimal orthogonal clustering and show, in Section 4.2.4, how the user can influence the clustering result. In Section 4.2.5, we prove that solving this model is NP-hard.

4.2.1 Almost Orthogonal Concepts

The data collected in today's applications, are often generated by different concepts which are mixed together. In an optimal setting, the concepts, described by subspaces, share no dimensions and we can clearly distinguish between them. If we identify a concept in the subspace S all other subspaces T, which share at least some dimensions $T \cap S \neq \emptyset$, are similar to it and we can prune them. T cannot characterize a different concept because a dimension $d \in S \cap T$ is already covered by the concept in S and hence T does not detect a novel concept in this scenario. Hence, all subspaces that are similar to S are excluded from further consideration by the identification of S. This can be formalized by:

$$\begin{aligned} coveredSubspaces_0(S) &= \{T \subseteq Dim \mid T \cap S \neq \emptyset\} \\ &= \{T \subseteq Dim \mid |T \cap S| > 0\} \end{aligned}$$

A concept with the relevant subspace T is orthogonal to a concept in S if $T \notin coveredSubspaces_0(S)$. The dimensions of T and S are disjoint and hence we can detect novel information in T. So our clustering model only has to identify clusters in subspaces which are orthogonal and prune the already covered subspaces.

However, this orthogonality definition is too restrictive for our clustering model. Many subspaces are prohibited for selection and hence the resulting clustering contains only low information. By definition, each dimension appears in at most one concept. However, overlapping concepts are useful and expected in real life scenarios, e.g., the attribute "gender" in a customer database could appear in multiple concepts. For subspace clustering, we need a relaxation of the orthogonality property.

A less hard restriction is realized by the idea of excluding lower dimensional projections of S. The subspace S is more meaningful for the representation of a concept than using the projections which contain fewer attributes. Hence, if we identify S as the relevant subspace for a concept, each projection is already

described by this subspace. The subspaces similar to S can be defined by:

$$\begin{aligned} coveredSubspaces_1(S) = \mathcal{P}(S) &= \{T \subseteq Dim \mid T \subseteq S\} \\ &= \{T \subseteq Dim \mid T \cap S = T\} \\ &= \{T \subseteq Dim \mid |T \cap S| = |T|\} \end{aligned}$$

By this definition, we can find overlapping concepts, e.g., characterized by $S_1 = \{1, 2\}$ and $S_2 = \{2, 3\}$. Neither of them is similar to the other concept and hence both of them could appear in the result set together. This definition is related to the maximality property in other subspace clustering approaches [AKMS07a, AKMS08a], resulting in the same problems. Even if two subspaces share a high fraction of dimensions, e.g., 9 out of 10, they represent different concepts. Thus, similarity of subspaces is not yet modeled in an adequate way so far.

Our model of almost orthogonal concepts integrates the advantages of both models. We allow overlapping concepts, but we also avoid concepts with too many shared dimensions. Thus, we only include (almost) orthogonal concepts in the result and obtain a flexible model by generalizing both definitions to:

$$coveredSubspaces_\beta(S) = \{T \subseteq Dim \mid |T \cap S| \geq \beta \cdot |T|\}$$

with $0 < \beta \leq 1$. For $\beta \to 0$ we get the first, for $\beta = 1$ the second definition.

The idea of our clustering model is to avoid the grouping of the same objects in similar concepts by several clusters. Given a cluster C we have to determine the set of clusters that are in similar concepts. Because we use orthogonal subspaces for the orthogonal concept detection, we can determine these clusters by checking if their subspaces cover the subspace of C. We call this set the concept group of C which can be formalized by the following definition.

Definition 4.1 *Concept group*
The concept group of $C = (O, S)$ with respect to a set of clusters $M = \{C_1, \ldots, C_n\}$ is defined as

$$conceptGroup(C, M) = \{C_i \in M \backslash \{C\} \mid S \in coveredSubspaces_\beta(S_i)\}$$

The concept group of $C = (O, S)$ contains all clusters that share at least a β-fraction of the dimensions of S. Checking the grouped objects O of C against the objects of its concept group is required to provide novel information within

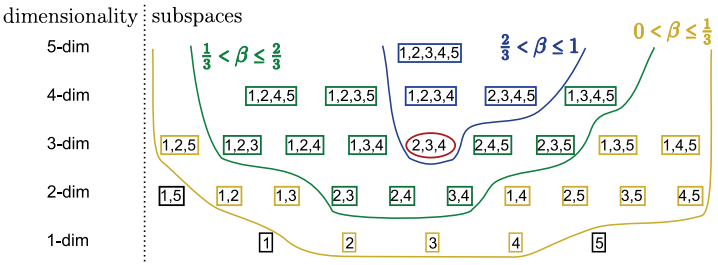


Figure 4.1: Concept group with variation of β

similar concepts. All other clusters, not in the concept group of C, do not need to be considered because they belong to other concepts. We permit such multiple concepts in our result.

Let us consider the Figure 4.1 where the selected cluster C is in the subspace $\{2, 3, 4\}$. For $\beta \to 0$ we have to compare C with all clusters in subspaces sharing at least one dimension. C has to group new objects w.r.t. these clusters because they all characterize similar concepts. The higher β, the less subspaces are considered as similar and hence the more concepts are possible in the final clustering. The choice of $\beta = 1$ results in comparing C only to higher dimensional clusters C', which project to the subspace of C. For example, the concept described by the subspace $\{1, 2, 3, 4\}$ subsumes the concept of C and thus C has to be checked against this subspace. Thereby, we see that the concept group is not symmetric but it tends to include more higher dimensional clusters. The concept group of a low dimensional cluster, that is in general less interesting, usually contains more clusters compared to the one of a higher dimensional cluster. Thus, for a low dimensional cluster it is more difficult to provide novel information and consequently to be included in the result set.

4.2.2 Global Interestingness

After defining the clusters which characterize similar concepts as C, we have to ensure that the cluster is interesting enough compared to these clusters. For our resulting clustering $Opt \subseteq All$, each cluster $C \in Opt$ has to fulfill this property. According to our motivation a cluster $C = (O, S)$ has to group new objects within the similar concepts. Hence, we use the coverage of objects as a criterion for interestingness. For a clustering $M = \{C_1, \ldots, C_n\}$ the coverage is defined as:

$$Coverage(M) = \bigcup_{i=1}^{n} O_i$$

A strict partitioning of the clusters in similar concepts, where we enforce that each object of C is in no other cluster, would be too restrictive. Even in similar concepts it might be possible for individual objects to be part in multiple clusters, e.g., a person attending classic and rock concerts for the example in Fig. 3.1. Therefore, we relax this property and calculate the relative fraction of objects which are not covered by other clusters in similar concepts w.r.t. the whole cluster size.

Definition 4.2 *Global interestingness*
Given a cluster $C = (O, S)$ and a set of clusters $M = \{C_1, \ldots, C_n\}$. The global interestingness of C with respect to M is

$$I_{global}(C, M) = \frac{|O \backslash Coverage(conceptGroup(C, M))|}{|O|}$$

First, we determine the clusters in similar concepts to the one of C and afterwards their objects are removed from O to obtain the newly covered objects of C. Only if $I_{global}(C, M)$ is larger than a given threshold α, the cluster adds sufficiently new information to this concept.

Figure 3.2 illustrates this interestingness check. Let us assume that M contains the clusters C_7 to C_{10} and possible further clusters in other subspaces (not within dimension 3). If we choose $C = C_{10}$, the concept group corresponds to $\{C_7, C_8, C_9\}$. The remaining clusters are not considered because they represent other concepts. C_{10} has to group new objects within the concept. However, most of the objects (29 out of 32) from C_{10} are already covered by the other clusters and, hence, the information obtained by C_{10} in this concept is small ($I_{global}(C, M) = \frac{32-29}{32}$). For a threshold $\alpha > \frac{3}{32}$, the cluster C_{10} is regarded as redundant with respect to M.

The user is able to control the required interestingness of a cluster by variation of α. If the fraction of newly clustered objects is smaller than α, we do not choose the cluster. For the extremal value $\alpha = 1$, the clusters in similar concepts must not overlap. For $\alpha \to 0$ a cluster is selected as long as not all objects are covered by other clusters. Consequently, a high overlap is possible.

An important aspect of this model is that the interestingness of a cluster is checked against several clusters within similar concepts. Unlike other models [AKMS07a, AKMS08a], that make only a pairwise comparison of the object coverage, in our model all clusters from a similar concept are considered at the same

time to evaluate the interestingness of the new cluster. If we did not check against several clusters, the cluster C_{10} in Fig. 3.2 would get a misleadingly high interestingness value. A pairwise comparison of C_{10} to C_7 or C_8 indicates a high fraction of newly clustered objects which is in fact not true.

Let us choose a clustering $M \subseteq All$. The global interestingness ensures that each cluster $C \in M$ results in an information gain within its concept by covering new objects. Varying concepts are possible in M and considered by the definition. Thus, the proposed properties for a good clustering, mentioned at the beginning of the section, are guaranteed.

Definition 4.3 *Orthogonal clustering*
The clustering $M = \{C_1, \ldots, C_n\}$ is orthogonal iff

$$\forall C \in M : I_{global}(C, M \backslash \{C\}) \geq \alpha$$

The clustering $M = \{C_1, C_2, C_7, C_8, C_9\}$ in our example from Fig. 3.2 is an orthogonal clustering, while the clustering $M \cup \{C_{10}\}$ is not. However, the proposed definition alone is not yet sufficient to determine an optimal clustering $Opt \subseteq All$. Several clusterings could fulfill the definition, e.g., the trivial clustering $M = \emptyset$. The user wants to get an overview of the clustering structure and seeks for the most informative clusters. We have to ensure that these clusters are selected.

4.2.3 Optimal Orthogonal Clustering

While the global interestingness $I_{global}(C, M)$ always rates the cluster C with respect to a clustering M, we now assess the interestingness of the cluster C on its own. This so called local interestingness should correspond to the user-specific notion of interesting clusters. Formally, we have to define a function I_{local} which maps each cluster C to the value $I_{local}(C)$. This function could include different aspects, as the dimensionality or the size of the clusters. A discussion of this function is presented in Section 4.2.4.

Both, the global and local interestingness, are used to define our optimal orthogonal clustering. With the global property, we ensure that only informative clusters within similar concepts are selected. At the same time, we want to maximize the sum of the local interestingness for the resulting clusters. By maximizing the local interestingness, we get the most interesting clusters but also as many interesting clusters as possible (taking the orthogonal clustering constraint into account).

Definition 4.4 *Optimal orthogonal clustering (OOC)*
Given the set All of all possible subspace clusters, a clustering $Opt \subseteq All$ is an optimal orthogonal clustering iff

$$Opt = \arg \max_{M \in Ortho} \left\{ \sum_{C \in M} I_{local}(C) \right\}$$

with

$$Ortho = \{M \subseteq All \mid M \text{ is an orthogonal clustering}\}$$

In Fig. 3.2, we show an overall example with $\alpha = 0.5$ and $\beta = 0.5$. The clustering $M_1 = \{C_1, C_2, C_7, C_8, C_9\}$ is a valid orthogonal clustering, because each cluster covers a sufficient amount of new objects within its concept. Although C_1 and C_9 contain similar objects, the overlap is permitted because different concepts are realized. The clustering $M_1 \cup \{C_{10}\}$, for example, is not valid, because as shown in our previous example $I_{global}(C_{10}, M_1) = \frac{32-29}{32} < \alpha$. Obviously, each subset of M_1 is also an orthogonal clustering but less informative than M_1. Hence, these subsets cannot be optimal clusterings. If we assume that the user is more interested in high dimensional clusters and chooses I_{local} accordingly, the sum $\sum_{C \in M_1} I_{local}(C)$ will be maximal out of all orthogonal clusterings. Another orthogonal clustering, like $M_2 = \{C_1, C_2, C_{10}, \ldots, C_{13}\}$ which contains the one-dimensional projections of the second concept, would, therefore, result in a lower sum value. As a consequence, M_1 is preferred over M_2 and M_1 is the optimal clustering in this example.

Our model provides a selection of only interesting clusters in different and novel concepts. An overwhelming result size is prevented. As we use subspace clusters in our model, the interpretabiliy of the result set and the identification of the relevant attributes for each concept are guaranteed. Unlike other orthogonal clustering models, we keep the original dimensions and we use them for the orthogonality check. We steer the cluster selection to orthogonal subspaces.

4.2.4 Local Interestingness and Cluster Definition

Before we present our local interestingness function, we set up our cluster definition. We use density-based clustering because it detects arbitrarily shaped clusters even in noisy data [EKSX96]. The idea is to define clusters as dense areas separated by sparse areas. The density $density^S(p)$ of an object p in a subspace S is the number of objects in its ε-neighborhood around it. To identify clusters based on this density, we follow the definition from [KKK04], with the

modification that the ε-range is adjusted according to the dimensionality of the subspace. Therefore, we adapt the optimal bandwidth for density estimation [Sil86] to our clustering model. The value of ε in a subspace with dimensionality d is $\varepsilon = \left[\frac{4 \cdot n}{3 \cdot \Gamma(1.5)}\right]^{\frac{1}{5}} \cdot \varepsilon_1 \cdot \left[\frac{d+2}{4 \cdot n} \cdot \Gamma(\frac{d}{2}+1)\right]^{\frac{1}{5}}$ where ε_1 denotes the ε-range in the 1d subspace, n the database size, and Γ the gamma function.

With the cluster definition, we can define our user-specific local interestingness function. Three main properties characterize a subspace cluster $C = (O, S)$ in our cluster instantiation. The dimensionality $|S|$, the size $|O|$, and the density. A very dense cluster shows small variation in the attribute values of the relevant dimensions and, hence, is more interesting than a sparse cluster. We use the mean density $\frac{1}{|O|} \sum_{p \in O} density^S(p)$ over all objects within the cluster for this criterion.

Maximizing all measures at the same time is in general not possible, e.g., low dimensional clusters are usually larger than high dimensional clusters. Therefore, our local interestingness function subsumes all measures and gives the user the flexibility to weight the measures dependent on the application. The local interestingness function used in our experiments is

$$I_{local}(C) = |S|^a \cdot |O|^b \cdot \left(\frac{1}{|O|} \sum_{p \in O} density^S(p) \right)^c$$

with $C = (O, S)$ and $a + b + c = 1$.

4.2.5 Proof of NP-Hardness

In this section, we prove the NP-hardness of our optimal orthogonal clustering problem (OOC). For this we reduce the NP-complete $SetPacking$ problem [GJ79] to our model, i.e., $SetPacking \leq_P OOC$. Given several finite sets O_i the $SetPacking$ problem seeks for the maximal number of disjoint sets.

Theorem 4.1 *Computing the optimal orthogonal clustering OOC (Definition 4.4) is NP-hard.*

Proof 4.1
We show that $SetPacking \leq_p OOC$.

A. Input mapping: *Each set O_i is mapped to the cluster $C_i = (O_i, \{1\})$. Furthermore we set $\beta \in [0, \ldots, 1]$, $\alpha = 1$ and $I_{local}(C) = |S|$ (cf. Section 4.2.4, $a = 1, b = c = 0$).*

B. OOC ***generates a valid*** $SetPacking$ ***solution:***

1) The concept group contains all clusters:

$conceptGroup(C, M\backslash\{C\})$
$= \{C_i \in M\backslash\{C\} \mid S \in coveredSubspaces_\beta(S_i)\}$
$= \{C_i \in M\backslash\{C\} \mid \{1\} \in coveredSubspaces_\beta(\{1\})\}$
$= M\backslash\{C\}$

2) Each orthogonal clustering M *contains only disjoint sets:*

M *is orthogonal clustering*
$\Leftrightarrow \forall C \in M : I_{global}(C, M\backslash\{C\}) \geq 1$
$\Leftrightarrow \forall C \in M : \frac{|O\backslash Coverage(conceptGroup(C,M\backslash\{C\}))|}{|O|} \geq 1$
$\Leftrightarrow \forall C \in M : |O\backslash Coverage(M\backslash\{C\})| \geq |O|$
$\Leftrightarrow \forall C \in M : O_\cap \bigcup_{C_i \in M\backslash\{C\}} O_i = \emptyset$

3) Opt *contains maximal number of such disjoint sets:*

$Opt = \arg\max_{M \in Ortho}\{\sum_{C \in M} I_{local}(C)\}$
$\Leftrightarrow Opt = \arg\max_{M \in Ortho}\{\sum_{C \in M} |\{1\}|\}$
$\Leftrightarrow Opt = \arg\max_{M \in Ortho}\{\sum_{C \in M} 1\}$
$\Leftrightarrow Opt = \arg\max_{M \in Ortho}\{|M|\}$

(2) and (3) $\Rightarrow$ Opt *is a valid* $SetPacking$ *solution* $\Rightarrow$ OOC *is NP-hard*

4.3 The OSCLU Algorithm

The optimal orthogonal clustering has global properties, which increases the computational complexity. As we have already proven, the problem is NP-hard and, hence, we cannot expect that an efficient algorithm exists. Furthermore, we cannot generate the huge set of all subspace clusters All in a first step and select the optimal subset afterwards. We develop an approximation algorithm (OSCLU), that incrementally adds further clusters to the result set. For an efficient calculation, we integrate the clustering process into the concept and cluster selection process. This means that not all clusters in all subspaces are generated but many subspaces are pruned based on already detected concepts/clusters. An important question is which subspaces should be clustered first and hence which clusters should be added at the beginning to the result set to prune many other subspaces.

Traditional bottom-up approaches, that start with the low dimensional clusters, are not useful for pruning based on our global interestingness criterion. As

already mentioned, the concept group of a cluster contains mainly higher dimensional clusters (cf. Fig. 4.1). Thus, a low dimensional cluster has to compare its object coverage against more clusters than a high dimensional cluster. A low dimensional cluster is more likely to be excluded from the result set than a high dimensional cluster. For this reason, we use a top-down approach to add clusters to the final clustering.

Our algorithm, summarized in Algorithm 4.1, comprises three major contributions to avoid clustering of all subspaces. First, we develop a ranking of the subspaces (all with the same dimensionality) without clustering them (lines 4-6). The ranking accounts for the similarity of the current subspace with already detected concepts. The greater the number of already detected similar concepts, the less interesting is the subspace. In a second step, the ranking considers the possibility for a good clustering in a subspace based on efficient estimation. After ranking the subspaces, we use the first subspace for clustering (line 8). If clusters were identified, we incrementally update the result set (line 10). We have to consider the global interestingness so that redundant clusters are not selected. Furthermore, a high local interestingness of the selected clusters should be ensured. Resorting the ranking and the possible pruning of further subspaces (line 11) based on the new clusters is performed in order to push novel concepts to the top. If all subspaces with the dimensionality dim are pruned or selected for clustering, we decrease the dimensionality to realize the top-down approach.

Algorithm 4.1: OSCLU (Orthogonal Subspace CLUstering)

```
1  result set M := ∅
2  find initial dimensionality dim                    (Sec. 4.3.3)
3  while dim > 0 do
4      rank and prune subspaces based on              (Sec. 4.3.1)
5          1) subspace orthogonality score
6          2) subspace quality score
7      while ranking not empty do
8          choose best subspace for clustering
9          if clusters found then
10             update result set M                    (Sec. 4.3.2)
11             resort ranking and prune
12     dim=dim-1
13 return result set M
```

As an additional step, we present an efficient method that approximately identifies the highest dimensionality (of a subspace) in which clusters are expected (line 2). This avoids to start our ranking in the full-space, where clusters are in general not present.

4.3.1 Orthogonal Subspace Selection

Clustering each subspace is not efficient since many subspaces can be pruned because of already detected, similar concepts. We use two techniques to rank subspaces without clustering. The aim is to cluster only interesting and orthogonal subspaces. In our first approach, we use the similarity of already discovered concepts for pruning and ranking. The greater the number of similar subspaces in the result set, the higher is the possibility that new clusters in the current subspace cover the same concept and, hence, provide no novel information. We define the orthogonality score of a subspace S w.r.t. the current result set M as

$$orthogonality\_score(S, M) = \\ |\{T \subseteq Dim \mid S \in coveredSubspaces_\beta(T) \wedge \exists (O, T) \in M\}|$$

The definition is similar to the concept group, but considers only the subspaces. The higher the score, the worse is a subspace, because many similar concepts are already in the result set. The orthogonality score is the first criterion for ranking. Furthermore, all subspaces with a score greater than $maxOrth$ are removed from the ranking. This parameter can be controlled by the user and intuitively defines how detailed a concept is analyzed.

During the algorithm, the result set M changes and hence the orthogonality score does so too. By this, only the most informative subspaces are ranked top and, hence, are clustered. The clustering is concentrated to the orthogonal and novel concepts.

Our second approach makes use of subspace search [CFZ99, KKKW03] for measuring the quality of subspaces. Usually subspace search is a stand-alone technique for identifying interesting subspaces. Each subspace is mapped to a quality value, where a high value corresponds to a high possibility for a good clustering structure. we use this technique within the clustering task. We guide our algorithm to cluster only the most interesting subspaces based on the calculated qualities. Therefore, our ranking is extended such that all subspaces with the same orthogonality score are ranked again based on these qualities. In to-

tal, our ranking concentrates not only on novel concepts but also on high quality subspaces.

The subspace search method within our framework is easily exchangeable and we can use techniques like RIS [KKKW03] or ENCLUS [CFZ99]. For efficiency reasons, we develop an own technique which is able to exploit our density-based cluster definition (cf. Sec. 4.2.4). To have clusters in a subspace, several objects must have a high density according to our density-based clustering model, i.e., for an object p the value of $density^S(p)$ is large. We use a strategy that randomly selects points and calculates their mean density. This method is efficient and a good indicator for the existence of clusters. Let $Seeds$ be the set of randomly selected points, the quality score is then defined as

$$quality\_score(S, M) = \frac{1}{|Seeds|} \sum_{p \in Seeds} density^S(p)$$

The higher the quality, the better the subspace. As for the orthogonality score, we introduce a minimum score $minQual$ that each subspace has to fulfill to be maintained.

4.3.2 Incremental Result Construction

After ranking the subspaces based on the two scores, we select the first one and cluster it according to our model. We get a list of resulting clusters New. We now have to check which clusters $C \in New$ should be included in our result set M. In a first step, we analyze the global interestingness of the new clusters. For each cluster $C \in New$, we calculate $I_{global}(C, M)$. We distinguish two cases.

If $I_{global}(C, M) \geq \alpha$, we directly add the cluster to M, i.e., the new result set is $M := M \cup \{C\}$. The cluster C adds sufficiently new information. By this, we ensure that in each step of the algorithm only informative clusters are selected. Please note that this procedure is a relaxation of Def. 4.3. We do not check the global interestingness of the remaining clusters in M which could be changed by selection of C. This recalculation would be too costly. However, due to our top-down approach, higher dimensional clusters are added first to M and these clusters are rarely removed by low dimensional clusters. Additionally, within the same dimensionalities, our ranking tries to rank the best subspaces on top and, hence, these clusters are selected first.

If $I_{global}(C, M) < \alpha$, we do not reject the cluster immediately but we perform an additional improvement heuristic. We want to maximize the local interest-

ingness in our model. Hence, we check if it is possible to remove some clusters from M such that C is afterwards globally interesting and the sum of the local interestingness is increased. The algorithm which decides if C is included and which subset of M should be removed is presented in Alg. 4.2. First, we rank the clusters from $conceptGroup(C, M)$ in decreasing order based on their local interestingness values. Second, we select the most interesting clusters which do not result in redundancy for C (set N). The clusters which induce the redundancy are stored in R. At the end, it holds that $I_{global}(C, M \backslash R) \geq \alpha$, i.e., C provides novel information with respect to the new set. If the local interestingness of C is greater than the one of R, it is better (for maximizing the interestingness) to select C and remove R from the result set M. The new result set is then $M := (M \backslash R) \cup \{C\}$. Otherwise, C is rejected and the set M remains unchanged.

Algorithm 4.2: Cluster selection procedure

```
⟨C_1, ..., C_n⟩ := ranking of conceptGroup(C, M)
N := ∅ //clusters inducing the redundancy of C
R := ∅ //clusters not inducing redundancy of C
for i:1...n do
    if I_global(C, N ∪ {C_i}) ≥ α then N := N ∪ {C_i}
    else R := R ∪ {C_i}
if I_local(C) > Σ_{C'∈R} I_local(C') then
    add C to M and remove R from M
```

Through the incremental result construction, we add only informative clusters to our set and additionally try to maximize the interestingness of all selected clusters.

4.3.3 Efficient Initialization

In general, full dimensional clusters are not identified in high dimensional databases. If we started our top-down approach in full-space, we would analyze many uninteresting subspaces which are filtered out by our quality score criterion. For an efficiency boost, we identify the first layer with interesting subspaces based on the idea of binary search. We start with the "half-dimensional" spaces (e.g., 5d spaces in a 9d database) and use our subspace search estimator to calculate the qualities. If we identify a subspace with sufficiently high quality, we directly jump up to the dimensionality between half and full-space (e.g., from 5 to 7). If

no interesting subspaces are found, we accordingly jump to lower dimensional spaces (e.g., 5 to 3). For this new dimensionality, we repeat the "check-and-jump" procedure (with a half jump range) until we identify the highest dimensionality with interesting subspaces. This corresponds to a binary search procedure.

Overall, our algorithm comprises three contributions to obtain a good approximation of the optimal orthogonal clustering. The binary search technique supports the top-down approach by an efficient initialization. The ranking of subspaces yields a preference of orthogonal and interesting subspaces. By recalculating the ranking, further subspaces can be pruned without clustering and novel concepts advance to the top. At last, the meaningful selection of new clusters to M results in an informative clustering. In the next section, we confirm this with an experimental analysis.

4.4 Experiments

We evaluate the quality and efficiency of the *OSCLU* approach compared to three variants of orthogonal clustering techniques (*Multi-View 1* and *Multi-View 2* proposed in [CFD07], and *Altern. Clus.* [QD09]), a recent non-redundant subspace clustering technique (*StatPC* [MS08]), and a projected clustering approach (*P3C* [MSE06]). For fair comparison, we use a recent evaluation framework [MGAS09], additionally reimplement both Multi-View approaches in this framework, and use the original implementation for the alternative clustering [QD09]. Furthermore, for all algorithms we tried to find the optimal parameter settings for each dataset.

In general, we perform our evaluation on data with multiple hidden concepts. For both, synthetic and real world data, we extend single concept data used in traditional clustering approaches such that each object is part of multiple concepts. Thus, for a high quality clustering, each object has to be detected in multiple clusters. While traditional clustering approaches are well suited for data with only one hidden concept, we compare our approach against recent techniques designed for multiple hidden concepts. For scalability experiments, we generate synthetic data following a method proposed in [KKK04, AKMS08a] to generate density-based clusters in arbitrary subspaces. In addition, our generator takes into account that objects can belong to multiple concepts. Thus, for each object, we concatenate attribute values of different subspace clusters to a higher dimensional space with multiple hidden concepts per object. Further on, we show the

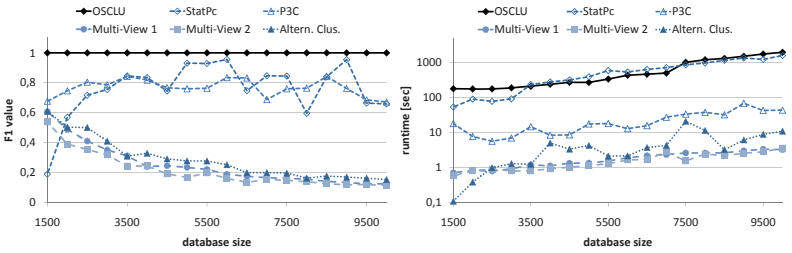
performance of OSCLU on two extended real world datasets (original *iris* and *liver disorders* are provided by the UCI repository [FA10]). We use the class labels in these datasets as one hidden concept of the data. In addition, we create multiple concepts per object by randomly concatenating objects of different classes, resulting in one high dimensional dataset.

To ensure comparability of evaluations, we measure runtimes on identical machines with 2.33GHz Intel XEON CPU, 2 GB of main memory and JAVA 1.6 runtime environment. Furthermore, for comparable quality measurements we use the $F1$ value that is used in evaluation of subspace and projected clustering [AKMS08a, MS08, MSE06, MGAS09]. In our case, it computes for each hidden cluster the harmonic mean of recall ("are all objects of the hidden cluster detected?") and precision ("how accurately is the cluster detected?") values, respectively. Therefore, each hidden cluster is evaluated against one of the detected clusters which provides the highest F1 value. The F1 value of the whole clustering is simply the average of the F1 values for each hidden cluster.

4.4.1 Scalability

Database size. In Fig. 4.2(a), we analyze the quality of the clustering results with respect to the database size. While increasing the number of objects, we keep the number of concepts fixed to three. We generate concepts with five relevant attributes such that, overall, we obtain a 15-dimensional data space. Our OSCLU algorithm yields the highest quality compared to all other algorithms, as we detect all hidden clusters in various concepts. The quality of OSCLU is independent of the database size and very robust. StatPC and P3C show good quality results, but also high fluctuating values, which cannot reach the quality of OSCLU. All three orthogonal clustering approaches, show only low and decreasing quality with respect to the database size. Their underlying k-means model tries to partition the data in each iteration of orthogonal cluster detection. Thus, it cannot cope with the fixed noise ratio in the data which is always assigned to some of the detected clusters and, hence, resulting in low quality clusterings.

The runtime with respect to the database size is presented in Fig. 4.2(b). The slopes of all curves are in the same range and the influence of the size on all algorithms becomes apparent. The two top quality approaches, our OSCLU model and StatPC, result in similar runtimes. Our redundancy checks and also our density-based model are very complex, but these aspects account for the high



(a) Quality of clustering (b) Runtime

Figure 4.2: Scalability w.r.t. database size

quality. The remaining algorithms are faster but we believe, that our runtime is still acceptable considering our high quality results. Especially with increasing concept number, as presented in the next experiment, our model outperforms all other approaches.

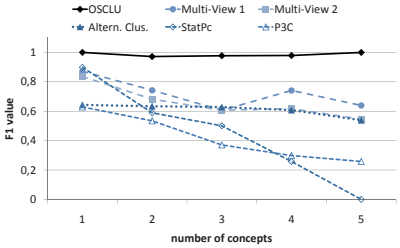


Figure 4.3: F1 vs. number of concepts

Number of concepts. The aim of our model is the detection of multiple concepts, which arise in real scenarios. Thus, in the next experiment, we analyze the performance of the algorithm by increasing the number of concepts hidden in a database. To scale the number of concepts, we use a simple dataset with only 1000 objects, as most of the algorithms showed comparable quality values in this range in the previous experiment. We vary the number of hidden concepts in Fig. 4.3 from 1 to 5.

We show that OSCLU is able to detect clusters even if objects cluster in multiple concepts. It shows high quality even for a high number of hidden concepts. While traditional clustering approaches aim at clustering single concept data, the alternative clustering approach is designed for two concepts and the multi-view

approaches should detect even more than two. However, even these approaches cannot compete with our model. For the subspace and projected clustering approaches, increasing the number of concepts makes it very hard to detect the hidden clusters. Especially the projected clustering approach P3C shows decreasing quality, as each object belongs to at most one concept. Overall, StatPC and P3C are not able to detect the multiple hidden concepts per object, while OSCLU yields very high clustering quality.

Noise percentage. In the previous experiments, we showed that we outperform subspace and projected clustering approaches as they cannot cope with multiple hidden concepts. Thus, in the following experiments, we focus on a more detailed comparison of OSCLU against the orthogonal clustering techniques detecting multi-view and alternative clusterings. First, we analyze the effect of noisy data especially for high concept numbers. For the next experiment, illustrated in Fig. 4.4, we generate data with five hidden concepts and vary the noise percentage. On such a difficult data setting, our OSCLU approach outperforms the other techniques. It can detect the clusters hidden in different concepts even in very noisy datasets. Both multi-view algorithms and the alternative clustering approach show, again, decreasing qualities.

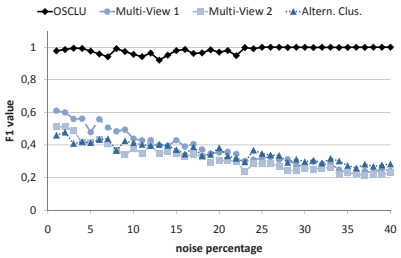
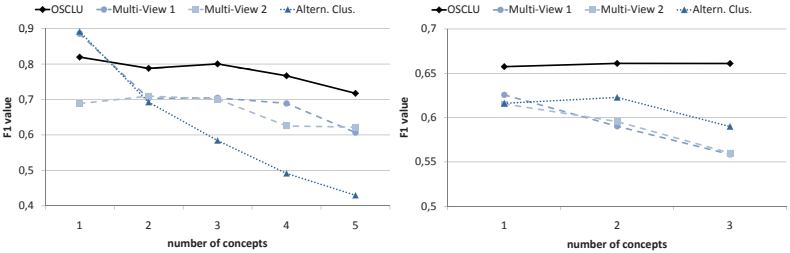


Figure 4.4: F1 vs. noise

4.4.2 Real World Data

As we aim at detection of multiple concepts, we focus our evaluation for real world data also on scalability w.r.t. number of concepts. We use single concept data from the UCI repository [FA10] and extend them to multi concept datasets. As described in the experiment set-up, similar to synthetic data, we can vary the complexity of datasets by including more and more hidden concepts. However,

in contrast to the previous experiments, we use real world data distributions for the single concepts. We evaluate the effect of variable concept counts on the clustering quality, as for an increasing number of concepts, it is more difficult for all algorithms to identify the hidden structure of the data.



(a) Dataset: multi-concept iris (b) Dataset: multi-concept liver disorder

Figure 4.5: Quality on extended real world datasets with increasing number of concepts

In Fig. 4.5 we show the clustering quality on the iris and the liver disorder dataset. For the very simple case of only one concept (original UCI datasets), the quality is high for all algorithms. However, for an increasing number of hidden concepts, the quality dramatically drops for all competing approaches. Especially, the quality of the alternative clustering approach drops with more than two concepts, as it is designed for up to two concepts only. OSCLU shows significantly better performance as it still achieves a high quality, outperforming the competing approaches for multiple concept data. Although we set for higher number of concepts the optimal parameter value k such that the number of found clusters corresponds to the number of hidden clusters, the competing approaches are not able to detect all hidden concepts. Thus, our OSCLU approach clearly outperforms all competing algorithms even for an increasing number of concepts per object.

4.4.3 Parametrization

Additional to the experiments that compare OSCLU to existing methods, we analyze the flexibility of our model. As presented in Section 4.2.2, the user can control the output by changing the required interestingness. In Fig. 4.6, we present

the variation of the parameter α which controls the interestingness of each cluster in the result set. As intended by this parameter, higher values of α increase the required interestingness and, hence, less clusters are in the result. By varying the α, parameter one can control the overall result set based on our global interestingness I_{global}. We include a cluster only if the fraction of its newly clustered objects is at least α (cf. Def. 4.3). Furthermore, our OSCLU algorithm is not only able to detect orthogonal concepts, but in addition it is very flexible by using a local interestingness I_{local}. It allows the user to control the output dependent on the application or the current interestingness.

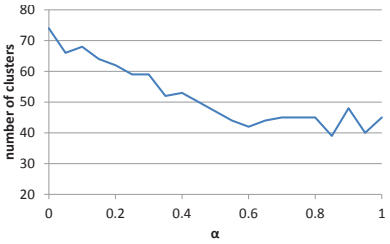


Figure 4.6: Variation of parameter α for the global interestingness

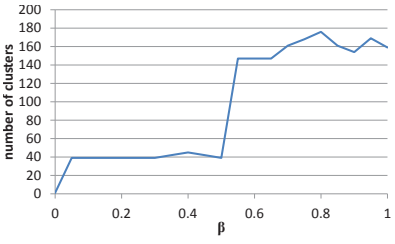


Figure 4.7: Variation of parameter β for the concept group

In Fig. 4.7, we examine the influence of the parameter β, which determines the similarity of two subspaces and, thus, the size of the concept groups, on the average result size. For small values of β, concepts are only dissimilar if they share nearly no dimension. Higher values for β accordingly allow for bigger overlaps between different concepts. Thus, the concept group of a cluster is biggest for $\beta = 0$ and smallest for $\beta = 1$ (cf. Def. 4.1). Since the global interestingness, i.e., the redundance, of a cluster is determined in relation to its concept group, we obtain an increasing size of the result set for increasing values of β (cf. Fig. 4.7).

4.5 Conclusion

In this chapter, we introduced the OSCLU (Orthogonal Subspace CLUstering) approach. It provides a general solution, independent of the chosen cluster definition, for detecting clusters in multiple views and overcomes major drawbacks of existing approaches in the detection of multiple concepts hidden in arbitrary sub-

space projections of the data. Our novel clustering model detects multiple concepts per object. It computes an optimal orthogonal clustering by ensuring non-redundancy and maximal interestingness of the resulting clustering. We show that our clustering model is NP-hard and propose an efficient approximative algorithm. We approximate the optimization problem by pruning similar subspaces, ensuring efficient cluster detection in only the orthogonal subspaces. Thus, our OSCLU approach is the first method for detection of multiple orthogonal concepts in subspaces of high dimensional data. Thorough experiments demonstrate that OSCLU clearly outperforms existing subspace clustering and orthogonal clustering algorithms, while automatically reducing the output to only the clusters of orthogonal concepts hidden in the data.

An important difference of OSCLU compared to multi-view approaches is its focus on clusters instead of views. For OSCLU, the views do influence but not completely determine the individual groupings, which also depend on the cluster definition. Therefore, OSCLU does not explicitly mine the underlying views but only their clusters depending on the subspace cluster definition. The identification of clusters that belong to a common view can be achieved in a post-processing step, for which we will present an interactive, visual approach in Chapter 15.

5

Detection of Alternative Subspace Clustering Concepts

THE goal of clustering is to detect new, hidden patterns in the data. In many cases, the users are already aware of certain patterns in the data. Simply rediscovering such existing knowledge is not of interest. Instead, users hope to find further, so far undiscovered patterns that reveal new insights of the data. The idea of alternative clustering approaches is to use the existing patterns to guide a clustering process towards new cluster constellations. In this chapter, we will present an approach to transfer this principle to subspace clustering. With just minor adaptions of the previously presented OSCLU approach, we will be able to incorporate a known subspace clustering into the clustering process and to steer the algorithm towards novel clusters in different subspaces.

5.1 Introduction

Traditional clustering and subspace clustering methods are not based on the assumption that there exists some prior knowledge about groupings in the data. However, we might already know some trivial or already detected groupings in the data. If the user is not satisfied with the existent knowledge, either because it does not meet her application needs, or because she assumes that there must exist further patterns in the data, then she aims for an alternative, yet comparable good clustering. In such scenarios the user is not willing to re-detect the already known clusters. As a general objective for recent alternative clustering techniques, it is important to acquire novel knowledge (not known in advance) by alternative clusters representing different views on the same database. The detection of such alternative clusters describing different views on each object is still an open challenge in recent applications.

In the previous Chapter 4, we presented the OSCLU approach, which is able to detect orthogonal concepts, i.e., differing clusters in orthogonal subspace projections of the data. In this chapter, we present how the OSCLU model can be easily extended for the task of alternative clustering. Given a (subspace) clustering as prior knowledge, the task of alternative (subspace) clustering is to detect further alternative groupings hidden in different views of the given database. For example, in sensor analysis one aims at detecting sensor groups showing similar measurements. Each sensor might be grouped in multiple alternative clusters. One object might be clustered due to its high temperature and low humidity measurements in the "hot and dry region" cluster, while the same object might be clustered in the "light region" cluster considering only the illumination attribute. Assuming these two clusters as given prior knowledge, further interesting alternative clusters might be hidden in the database, e.g., a grouping of sensors representing a "dark and humid region". Such an alternative cluster might be of great importance in addition to the given two clusters. However, there might also be some trivial useless clusters, like objects clustered in both a "dry region" and in a "hot region". Obviously these two clusters only provide redundant information to the given "hot and dry region" cluster. As illustrated in this toy example, the detection of alternative clusters is of great importance, especially in recent applications where clusters are hidden in any possible attribute combination.

In general, we detect clusters hidden in subspace projections of the database to identify multiple views on the data. However, several new challenges arise for

the research area of alternative subspace cluster detection. As one searches for clusters in arbitrary subspaces, each cluster might be detected in multiple redundant views. Similarly, the knowledge of already given clusters might be repeated in similar subspace clusters. In both cases the OSCLU approach provides a nice solution for detecting new clusters not yet detected by other subspace clusters and not yet represented by the given clusters. Thus, our main contributions include:

- Detection of alternative subspace clusters
- Non-redundant clusters (dissimilar to each other)
- Alternative clusters (dissimilar to given clusters)

5.2 Related Work

Recent extensions of traditional clustering techniques try to detect clusters that are alternative to a given, known clustering. The techniques of [CFD07, DQ08, QD09, BB06, DB13b] base on a given clustering and iteratively transform the data space to force the underlying traditional clustering algorithm to find new, alternative clusters. Other techniques, like [GH05], follow the idea of using the conditional information bottleneck approach to find alternative clusterings. All these techniques are not able to detect clusters hidden in arbitrary subspace projections of the data and consider in each step only one fixed space. Furthermore, their input clustering has to be a partition of the data in a fixed space, whereas we allow a subspace clustering as input, which has multiple locally relevant subspaces. Only two of these approaches briefly refer to subspace clustering.

Although the method presented in [QD09] searches for clusters in the full-space, it can be adapted to handle subspace clusters (of a single fixed subspace) as input by simply setting the values in the relevant attributes to zero. These dimensions therefore loose their influence in the following iterations. However, this complete elimination of covered attributes leads to an orthogonal subspace, which is a too strong restriction for the choice of relevant dimensions.

The approach in [CFD07] has originally not been introduced to find alternatives for a given clustering but can easily be adapted by replacing the initial k-means clustering through the known clustering solution. Since this approach projects the data into an orthogonal space to find alternative clustering views, it is also not a subspace clustering and suffers from the problems already mentioned for [QD09].

Although these methods are, with large restrictions, able to find clusters in a (fixed) subspace, they are mostly not aware of the relevant subspaces and can therefore not annotate them to the clusters. The reason why objects group in a certain manner, however, originates from the respective subspace, which makes the relevant attributes an essential aspect to the clusters information. Furthermore, they are not able to guarantee that a new clustering solution is truly alternative if we consider multiple known clusterings. As they are designed to consider only partitionings of the data, it is not possible to integrate a subspace clustering, where clusters overlap object-wise, as knowledge base.

5.3 Alternative Subspace Clusters

In this section, we describe our model for finding alternative subspace clusterings. To achieve this goal, our model adapts techniques of the OSCLU model (Chapter 4) that finds orthogonal subspace clusterings in the data. With our novel method, however, we specifically address the problem of finding an alternative subspace clustering given a previously known subspace clustering. Thereby, we achieve that the user can steer the clustering algorithm to patterns not yet detected and the generation of already known clusters is prevented.

In general, a subspace cluster $C = (O, S)$ is a set of objects $O \subseteq DB$ and a set of dimensions $S \subseteq Dim$. The objects O are similar within the relevant dimensions S, while the dimensions $Dim \backslash S$ are irrelevant for the cluster. In Fig. 5.1, the cluster C_1 corresponds to a 2-dimensional cluster, while C_2 is a 1-dimensional one. The input of our model is an already known subspace clustering $Known = \{K_1, \ldots, K_m\}$ where K_i is a subspace cluster. The task is to identify another subspace clustering within the database that differs from the given one. For our example in Fig. 5.1, we assume $Known = \{C_1, C_2\}$. A possible alternative solution is $\{C_4, C_5, C_6\}$. This solution is interesting because we detect clusters in novel subspaces of the database. We designed our model to be independent of the actual cluster definition, i.e., we assume a set $All = \{C_1, \ldots, C_k\}$ of possible subspace clusters is given (cf. Sec. 5.3.3 for our instantiation). In Fig. 5.1, we assume $All = \{C_1, \ldots, C_7\}$. The set All is also a subspace clustering; however, it is not an (good) alternative to $Known$. Beside other criteria, this set contains clusters very similar to clusters in the input clustering. Thus, the overall goal of our model is to select a meaningful subset $Res \subseteq All$ as the result presented to the user.

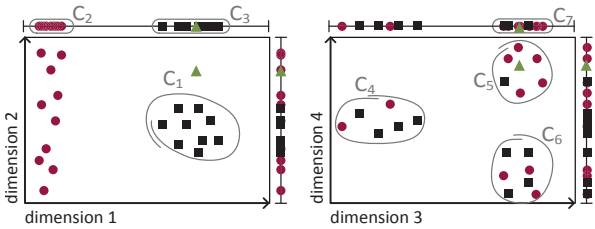


Figure 5.1: Exemplary subspace clusters

Problem statement: Given an already known subspace clustering $Known = \{K_1, \ldots, K_m\}$*, the aim of* alternative subspace clustering *is to determine a meaningful subset* $Res \subseteq All$ *of all possible subspace clusters* $All = \{C_1, \ldots, C_k\}$*, such that* Res *differs from the input clustering.*

In the following, we discuss the criteria a meaningful alternative clustering solution has to fulfill and we define the overall result.

5.3.1 Valid alternative subspace clustering

Given the clustering $Known$, we want to detect a valid alternative clustering Res. Which properties must hold true for Res to be a *valid* alternative? Apparently, each cluster $C \in Res$ should considerably deviate from the clusters in $Known$. The cluster C should provide us with novel knowledge. For subspace clustering, we have two possibilities to realize a deviation to already known clusters. First, our novel subspace cluster comprises a "different" (i.e., novel) subspace or, second, it covers "different" (i.e., novel) objects in already known subspaces. Thus, a cluster C is not a valid alternative if the subspace as well as the objects are already clustered.

Alternative w.r.t. subspaces. If the subspaces of two clusters differ substantially, both are interesting, even if their clustered object sets are nearly identical. Different subspaces mean different relevant attributes and, hence, a valid alternative. However, it is problematic to deduce that a cluster $C = (O, S)$ is a valid alternative to $C' = (O', S')$ only based on the fact $S \neq S'$. It is a well known observation in the area of subspace clustering that similar object groupings appear in very similar subspaces several times: this is one aspect of the redundancy problem in subspace clustering [MAG+09b, MS08, AKMS08b]. Considering for example the clusters C_1 and C_3 in Fig. 5.1, their subspaces are unequal but very

similar. Thus, a grouping of similar objects is expected. For our task of finding alternatives, we have to ensure that the subspaces of our novel clusters differ to a high extend compared to the ones of the given input clusters. If a cluster $C \in Res$ and a cluster $K \in Known$ have highly deviating subspaces, we, thus, do not have to enforce deviating object sets for these two clusters: C is already a good alternative with respect to the single cluster K. We use Definition 4.1 of a $conceptGroup$ of the OSCLU model and define the subset of clusters of $Known$ that are already different enough due to their subspaces:

Definition 5.1 *Clusters in alternative subspaces.*
Given a cluster $C \in Res$, the subset of clusters of $Known$ that belong to an alternative subspace w.r.t. $C = (O, S)$ is defined by

$$InAltSubsp(Known, C) = \{(O_i, S_i) \in Known \mid |S \cap S_i| < \beta \cdot |S|\}$$

with $0 < \beta \leq 1$.

Definition 5.1 and Definition 4.1 are similar in that $InAltSubsp(Known, C) = Known \backslash ConceptGroup(C, Known)$. If the fraction $|S \cap S_i|$ of the joint dimensions compared to all dimensions of C is small enough, the clusters represent different concepts and, hence, alternative information of the data. Thus, to decide whether C is a valid alternative to all clusters in $Known$, we can already neglect all clusters contained in $InAltSubsp(Known, C)$. In Fig. 5.1, we get $InAltSubsp(Known, C_4) = \{C_1, C_2\}$, because all of the input clusters were detected in completely alternative subspaces. For C_3 and with $\beta = 0.5$, however, we get $InAltSubsp(Known, C_3) = \emptyset$.

Keep in mind that this relation is not symmetric. Assuming an input cluster K from $Known$ in the subspace $\{1, 2, 3\}$ and a novel identified cluster C in the subspace $\{2, 3, 4, 5, 6\}$, then $K \in InAltSubsp(Known, C)$ for $\beta = 0.5$. The clusters share just two dimensions, which is significantly smaller compared to all five dimensions of C. Thus, C is already an alternative to K because there are enough new dimensions in C to provide novel information. However, assuming C is in $Known$ and K is the newly identified cluster, then $C \notin InAltSubsp(Known, K)$. There are still two common dimensions but these are now compared to just the three dimensions of K. With respect to the subspace, K is not an alternative to C because K mainly has dimensions already being included in C.

Alternative w.r.t. objects. For the possible case of $InAltSubsp(Known, C) \neq Known$, we have given some clusters that were detected in subspaces similar to C. Thus, for these clusters, we have to ensure a grouping of different objects

compared to C. The other clusters are already neglected because of deviating subspaces. Analogously to Definition 4.2 of the OSCLU model, we use the coverage of objects as a criterion for a deviating object representation. Given a cluster C, the set of objects already covered by $Known$ is defined as:

$$Covered(Known, C) = \bigcup_{(O,S)=K \in Known} \{O \mid K \notin InAltSubsp(Known, C)\}$$

If the covered objects of $C = (O, S)$ sufficiently differ from the covered objects of the clusters in $Known$, the cluster C is a valid alternative.

Definition 5.2 *Valid alternative subspace clustering*
Given a cluster $C \in Res$, $C = (O, S)$ is a valid alternative cluster to $Known$ iff

$$\frac{|O \backslash Covered(Known, C)|}{|O|} \geq \alpha$$

with $0 < \alpha \leq 1$.
Given a clustering $Res \subseteq All$, Res is a valid alternative clustering to $Known$ iff all clusters $C \in Res$ are valid alternative clusters to $Known$.

In Fig. 5.1, the set $Covered(Known, C)$ already contains nearly all objects of C_3. Since the fraction of novel objects of C_3 is very low, e.g., for $\alpha = 0.4$ the cluster is not a valid alternative (less than 40% of the objects are novel). However, choosing $Known = \{C_2, C_5\}$, cluster C_3 is a valid alternative. C_5 is located in a different subspace, hence, irrelevant for C_3, and C_2 covers different objects.

5.3.2 Optimal alternative subspace clustering

With Definition 5.2, we are able to find meaningful alternatives to a given input clustering. However, among the set of possible clusterings, we can find multiple valid alternatives. For $Known = \{C_2, C_5\}$, e.g., the results $Res = \{C_3, C_4, C_6\}$, $Res' = \{C_1, C_3, C_4, C_6\}$, or $Res'' = \{C_3, C_4, C_7\}$ would be among the valid alternative clusterings for $Known$. Since these solutions are not equally interesting for the user, the task is to find the most interesting alternative clustering. The main criteria for the interestingness of a clustering are the degree of redundancy among the clusters and local characteristics of the clusters.

Redundancy. Since for subspace clustering a partitioning of the objects is not enforced, the solution could contain very similar clusters. The solution $Res' = \{C_1, C_3, C_4, C_6\}$ is a valid alternative, although the clusters C_1 and C_3 are very

similar to each other and, therefore, introduce redundancy into the result. OSCLU provides an elegant and easy way to solve this problem through the Definition 4.3 of an orthogonal clustering which is related to the Definition 5.2 of a valid alternative clustering. According to Defintion 4.3 Res' would not be an orthogonal clustering since $I_{global}(C_1, Res' \backslash C_1) \not\geq \alpha$. Contrarily, the set $Res = \{C_3, C_4, C_6\}$ is not only a valid alternative to $Known$ but also an orthogonal and, thus, redundancy-free clustering.

Local interestingness. Although by avoiding redundant clusterings we reduce the number of possible clustering solutions, many clusterings are still possible. For example, both sets $\{C_3, C_4, C_6\}$ and $\{C_1, C_4, C_6\}$ are valid alternatives and orthogonal clusterings. To decide between those solutions, we utilize the idea of OSCLU to take local characteristics of the clusters into consideration. For different applications different local characteristics can be of interest. Among others, possible choices can be the density, size, extension, or dimensionality of a cluster. According to these characteristics, each cluster is annotated with a certain interestingness value. By selecting those clusters that maximize this interestingness, we get the desired result. For the OSCLU approach, a local interestingness function I_{local} maps each subspace cluster C to the interestingness value $I_{local}(C)$. Our instantiation is presented in Section 5.3.3. The overall interestingness of a clustering Res is obtained by summing up the individual values of each cluster: $quality(Res) = \sum_{C \in Res} I_{local}(C)$. Assuming higher dimensional clusters to be more interesting in our example, the quality of $\{C_1, C_4, C_6\}$ is higher than the one of $\{C_3, C_4, C_6\}$.

Accounting for the redundancy and the local interestingness, we are now able to define our overall clustering solution:

Definition 5.3 *Optimal alternative subspace clustering.*
Given a previously known subspace clustering $Known$ and the set of all possible subspace clusters All, a clustering $Res \subseteq All$ is an optimal alternative subspace clustering iff

- *a) Res is a valid alternative to $Known$*
- *b) $\forall C \in Res : \{C\}$ is a valid alternative to $Res \backslash \{C\}$*
- *c) Res is the most interesting clustering, i.e., $\forall Res' \subseteq All$ that also fulfill a & b: $quality(Res) \geq quality(Res')$*

With this new model, we are able to determine a subspace clustering result that differs from the input clustering to a high extent: either by representing

novel objects or by comprising novel subspaces. At the same time, we avoid generating redundant clusters for the result, focusing again on deviating subspace clusters.

5.3.3 Instantiation and algorithm

Instantiation. The instantiation of our ASCLU approach (Alternative Subspace Clustering) is identical to that of OSCLU, which allows us to reuse the algorithm designed for OSCLU. The cluster definition is based on the density-based clustering paradigm because it allows for arbitrarily shaped clusters even in the presence of noise [EKSX96]. A cluster is determined via dense areas in the data space [KKK04]. As in OSCLU, the density $density^S(p)$ of a point p in subspace S is determined by the cardinality of its ε-neighborhood where the variable ε is adjusted to the dimensionality of the subspace. The local interestingness function I follows the definition of OSCLU and incorporates the dimensionality, size, and density of the corresponding subspace cluster $C = (O, S)$: $I_{local}(C) = |S|^a \cdot |O|^b \cdot \left(\frac{1}{|O|} \sum_{p \in O} density^S(p)\right)^c$ with $a + b + c = 1$.

Brief overview of the algorithm. The OSCLU model was proven to be NP-hard. Since for $Known = \{\}$ the ASCLU model corresponds to the OSCLU model, we cannot expect that an efficient algorithm, exactly solving our model, exists. Instead, we slightly adapt the approximation algorithm of OSCLU that avoids generating the set of all possible subspace clusters by pruning several subspaces based on already detected patterns and using the knowledge of the input clustering. We incrementally add clusters to the current result set Res and we possibly refine this set if better clusters are detected. Technically, we use a top-down approach starting in high-dimensional spaces and traversing the subspace lattice in breadth-first order. During this traversal, the subspaces with the same dimensionality are processed in the order of their possible benefit for the clustering result. Subspaces that are highly different to subspaces already present in the input clustering $Known$ and different to the ones in the current result Res set are analyzed first. Too similar subspaces are pruned. The best ranked subspace is analyzed for its clustering structure using the density-based clustering model.

If clusters are identified in a certain subspace, we check if these clusters can be added to the set Res based on two criteria: First, the novel cluster C needs to be a valid alternative to $Known$. Second, the cluster C must not introduce redundancy into the current result set Res. If both criteria are fulfilled, we can

directly add this cluster to the result, i.e., $Res = Res \cup \{C\}$. If C is an alternative to $Known$ but redundant w.r.t. Res, clusters $M \subseteq Res$ have to exist that are responsible for the (current) redundancy. If we removed M from Res, we could add C to the result. To maximize the interestingness of our clustering solution, we determine the values $quality(Res \setminus M \cup \{C\})$ as well as $quality(Res)$ and we select the more interesting clustering. After updating the result set, the order of the subspaces, not yet analyzed, is potentially adapted. Overall, we steer our algorithm only to those subspaces where alternative clusters are expected and we avoid analyzing all subspaces.

5.4 Experiments

In this section, we evaluate the quality of the ASCLU (Alternative Subspace Clustering) approach and investigate whether it can provide a reasonable and non-redundant subspace clustering compared to a given set of input subspace clusters. For this goal, we start by applying ASCLU to synthetic data to get a better intuition of the main principle. The generation method for synthetic datasets corresponds to the one used in OSCLU. It produces density-based clusters in arbitrary subspaces, where each object can belong to multiple clusters with differing relevant subspaces. This generation method takes into account that objects can be relevant for several clusters in multiple views.

Furthermore, we demonstrate the performance of ASCLU on two real world datasets (iris and pendigits) provided by the UCI repository [FA10]. Since the motivating assumption for alternative clustering is the presence of multiple views in the data, traditional class-based evaluation, where each object is assigned to exactly one class, is not reasonable in this case. We, therefore, examine clustering results for the pendigits dataset visually, similar to [CFD07]. For the iris dataset, we concatenate the original data with random permutations of itself, which results in one high dimensional dataset containing several views: the concatenations. For the quality assessment we use the F1-value, as it can handle overlapping clusters and classes and is used for evaluation of subspace clusters [AKMS08b, MS08].

Experiments on synthetic data The first experiment serves to examine the ability of ASCLU to calculate a real alternative Res to a given clustering $Known$. An alternative clustering should yield new information compared to the given

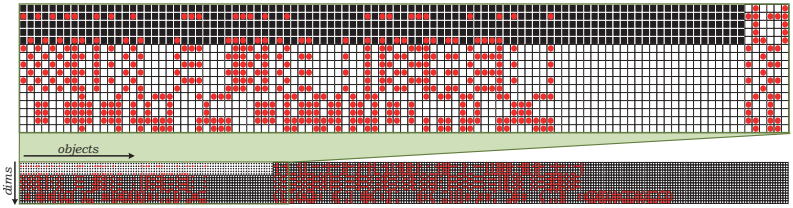


Figure 5.2: Data matrix for synthetic data with given clustering (black) and alternative clustering (red)

clustering. This can be characterized by clusters that either group similar objects compared to the given clusters but in different subspaces or simply group other objects than the given ones. A small synthetic dataset with 300 objects and 16 dimensions, where 16 clusters are hidden in five different subspaces, allows an easy visual examination. Fig. 5.2 depicts a representation of the data matrix, where the given clusters (black boxes) and all clusters found by ASCLU (red circles) are plotted. Each column of the matrix represents a database object, each row represents a dimension. For a clear presentation, the objects and dimensions have been permuted, such that the given clusters and several categories of new information types become apparent. The black rectangular area represents the previously known information. New clusters should preferably avoid this area of given clusters and concentrate on new information. The black area is only sparsely populated with circles, which indicates that the newly found clusters do not provide the same information. To gain new knowledge, a cluster has to cover a sufficient amount of new objects or/and different subspaces. Fig. 5.2 also shows that ASCLU does not block the given cluster area completely for the new clustering, like other approaches do for dimensions [QD09], but allows for clusters to overlap regarding dimensions or objects of the given clusters. The potential information content of the new clustering is, thus, extended.

Experiments on UCI datasets In the following, we focus on the effectiveness of ASCLU for real world datasets. We start using the pendigits dataset to show that ASCLU reveals new patterns compared to some given ones. The pendigits dataset is very rewarding for clustering analysis, since cluster results are very descriptive and visualizable. This dataset is suitable to examine, whether ASCLU is able to find valuable alternatives for given subspace clusters in real world data. As input for ASCLU, we use three clusters: digit 0 and digit 6, each clustered

in the first 3 xy-coordinates, and digit 9 clustered in the last 3 ones. Given this input, ASCLU determines subspace clusters with deviating properties and thus novel information. The first indicator is that the given three digits appear in less clusters than the other digits, which shows, that ASCLU is more interested in the unknown (not given) digits. A second observation is that the given digits appear mainly in clusters that cover nearly all digits, thus representing novel object groupings. In Fig. 5.3 such an identified cluster is illustrated, where all digits have similar values for the marked y-coordinates. Only in very few clusters the three digits are again clustered individually; though, these clusters yield novel attribute information of the digits.

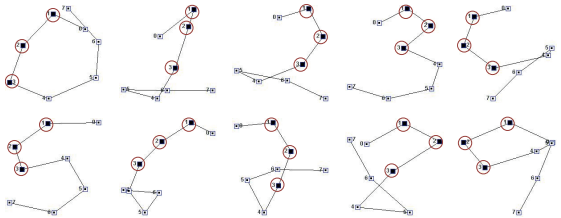


Figure 5.3: Alternative subspace cluster for pendigits

The next experiment, on the iris data, evaluates the influence of the input clustering on the quality of ASCLU's results. As described, we extend the iris dataset to multiple views per object. This way, we are able to determine a subspace clustering as input, which will in this case be the first view. We, therefore, only consider the classes of the other views, the latter concatenations, as ground truth to compare the results with. In Fig. 5.4, we compare the results of ASCLU with and without this first view as input clustering for three different datasets, which differ in the number of concatenated views. The results show, that a high quality input clustering, like one view in this example, has a positive effect on the quality of the alternatively found clusters in the other views. This effect is explainable by the fact that due to the given clusters, ASCLU already excludes several clusters with similar informations to avoid redundancy. These avoided similar clusters do, obviously, not belong to the ground truth and do often lead to the redundancy and thus the exclusion of valuable clusters in other views. As Fig. 5.4 shows, this effect is best traceable if there is only one view besides the given one. Nonetheless, the given information has also a positive effect on more than just one view.

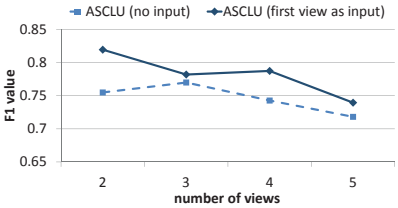


Figure 5.4: Quality of ASCLU on the iris dataset

5.5 Conclusion

In this chapter, we proposed a method that extended the principles of the previously presented OSCLU approach (Chapter 4) for detecting an alternative subspace clustering to a given input clustering. In contrast to previous approaches that determine alternative groupings, we specifically consider the relevant dimensions of each subspace cluster to identify different views within the data. Besides generating deviating clusters compared to the given input clustering, our model ensures that each resulting cluster provides novel knowledge by pruning redundant results. The experimental evaluation confirms that our model successfully detects meaningful alternative subspace clusters based on the given input clustering. Analogous to the OSCLU approach, also ASCLU does not explicitly mine views but focuses on clusters. Accordingly, also for ASCLU a separation of clusters according to views needs to be done in a post-processing step, such as the one proposed in Chapter 15.

Part III

Transfering Subspace Principles to the Multi-View Clustering Paradigm

It is the theory which decides what can be observed.
Albert Einstein

6

Introduction to Simultaneous Multi-View Clustering in Subspaces

BOTH techniques presented in Part II are aiming at transferring the principles of multi-view clustering for achieving dissimilar clustering results to the subspace clustering paradigm. The proposed approaches present general solutions for detecting clusters hidden in different attribute subspaces representing views on the data. The result in each case is, however, a single set of clusters, such that the views and the association of clusters to views are not determined explicitly. As we will see in Chapter 15, a post-processing of such subspace clustering results to obtain views and according alternative clusterings is not trivial and requires interaction with a domain expert. In this part, we will present two approaches to overcome these limitations by integrating the cluster definition and the view detection into a single model.

As already argued in the previous chapters, the detection of multiple clustering solutions in just a single space has limitations. Finding truly novel clustering structures by just considering a single data distribution is not very promising. Instead different characteristics of the data have to be highlighted to enable different views on the data. If we cluster, e.g., movies, we will easily achieve a different grouping based on their 'genre' compared to a clustering based on attributes like 'location', 'cast', 'budget', or other characteristics. As discussed in Chapter 2, some approaches already consider different data representations for the task of multi-view clustering [CFD07, DQ08, QD09, NDJ10, DB13b]. However, most of these techniques consider data distortions leading to non-axis-parallel subspaces, e.g., based on PCA transformations, which are difficult to interpret semantically. By considering axis-parallel subspaces, we have a direct indication which data characteristics led to the observed cluster structures, which supports the semantic reasoning. If we, e.g., find a very compact cluster for the characteristics 'cast' and 'budget', which represents high-budget movies with certain actors which have no

other joint characteristics, we might start reasoning whether these are VIP actors whose salaries are causing the high expenses.

While in the next Part IV, we will use the subspace clustering principle to discover multiple clustering views iteratively, for the approaches proposed in this part, we will focus on the simultaneous generation of multiple clustering alternatives. A simultaneous detection of all hidden clustering alternatives has the advantage that the global interestingness, i.e., quality and diversity of *all* clusterings, can be optimized. Iterative approaches, instead, greedily choose the best available clustering in each iteration based on the previous knowledge. While the first detected clusterings probably will have a high clustering quality, in subsequent iterations bad clusterings might be preferred for the sake of diversity. Since in addition to the mere partitionings, we want to learn the according clustering perspectives, i.e., subspaces, for each clustering, the simultaneous consideration of all clusterings and their subspaces is beneficial as well. Not only the different clusterings can help to refine each other, also the chosen subspaces for each clustering, can be influenced by those of others. Although subspaces of different clusterings might share some characteristics, it is unlikely that highly overlapping subspaces support clusterings which are highly different. If the proportion of common characteristics is too high, the two subspaces will contain a very similar distribution of the data, such that truly deviating clusterings are not to be expected. Taking this assumption into consideration, the subspaces can help each other to define their clustering 'profile' more precisely.

An important criterion for the new approaches is furthermore, that the detection of an arbitrary number of alternative clusterings should be possible in order to detect all hidden concepts in the data. For the methods that we will present in the following Chapters 7 and 8, we use probabilistic generative models to solve the problem of finding multiple alternative clustering solutions in subspace projections of the data. These models can be nicely depicted by graphical models and capture the assumed causal process by which the data has been generated. In our case, each alternative clustering will be represented by a multivariate mixture distribution for the relevant attributes. For the approach in Chapter 7, we will focus on transferring the principles of subspace clustering to the paradigm of multi-view clustering. Here, we will explicitly consider local noise dimensions of clusters as well as overlapping subspaces for the different clustering views. For the second approach, that we will present in Chapter 8, we will simplify the considered generative model in order to focus on the integration of user constraints.

Often the user has some prior belief about the clustering structure and wants to guide the clustering process in a certain direction. So-called semi-supervised clustering techniques have shown to be able to drastically improve the clustering quality if such prior knowledge of the user is used as support. With our SMVC approach of Chapter 8, we want to examine a possibility to integrate such prior user knowledge into the complex clustering process of multi-view clustering in subspace projections.

7

Multi-View Clustering Using Mixture Models in Subspace Projections

In this chapter, we present a Bayesian framework to tackle the problem of simultaneous multi-view clustering in subspace projections of the data. We provide multiple generalizations of the data by using *multiple mixture models*. Each mixture describes a specific view on the data by using a mixture of Beta distributions in *subspace projections*. Since a mixture summarizes the clusters located in similar subspace projections, each view highlights specific aspects of the data. In addition, our model handles *overlapping views*, where the mixture components compete against each other in the data generation process. For efficiently learning the distributions, we propose the algorithm MVGen that exploits the principle of iterated conditional modes and uses Bayesian model selection to trade-off the cluster model's complexity against its goodness of fit. With experiments on various real-world datasets, we demonstrate the high potential of MVGen to detect multiple, overlapping clustering views in subspace projections of the data.

7.1 Introduction

Mixture models have proven to be well suited for adequately modeling and learning the characteristics of complex probability distributions of given observations in various applications [MK08]. In particular in the presence of an underlying clustering structure, multivariate mixture models are widely used as a compact representation of the data's distribution. Given a parametric family $\mathcal{K}$, e.g., the set of all Gaussian distributions, a mixture model describes the data by a set of components (each selected from $\mathcal{K}$) and a set of mixture weights. Intuitively, each component represents a cluster (more precisely: its distribution of the attribute values) and the mixture weight represents the number of objects belonging to this cluster.

Key to a reasonable data representation is an appropriate modeling of the underlying data structure. Traditional mixture models work with only a single mixture distribution, i.e., each observation is assumed to follow a single component's distribution. However, as the research areas of subspace clustering [KKZ09] and multi-view clustering [MGFS10] have taught us, for many data collections multiple, differing aspects of the observations are captured. This aspect has already been touched by approaches like [SJR10, FB08, BKG+05], that allow for a mixed membership in different components. Thus, they realize an overlapping clustering and, e.g., allow for a movie to participate in the 'humor' as well as in the 'action' genre [BKG+05]. These approaches still only realize a single clustering (i.e., one mixture model) and, therefore, are able to present only one view on the data, e.g., the view 'genre' for the movie example. For many scenarios, however, a more complex clustering structure, where different views on the data (i.e., considering different characteristics of the observations) reveal different clustering structures, has to be expected [NDJ10, CFD07, QD09]. Movies cannot only be clustered according to their 'genre' but also based on 'location', 'cast', 'budget', or other characteristics. Since data is rarely collected pursuing only one defined goal, the multi-view hypothesis is very likely for various databases, e.g., customer data, sensor data, biological records but also for data with various heterogeneous characteristics, like images or multimedia in general.

Just summarizing the data by a single global view, which considers all characteristics simultaneously, does not do such data justice. Instead, a generalization of the data by a mixture model *for each view* and its specific characteristics, reveals more insight in the data. Given the toy example in Fig. 7.1, we can easily

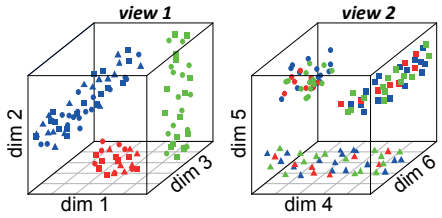


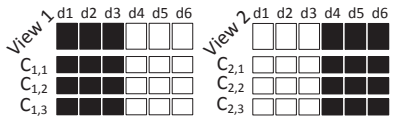
Figure 7.1: Example for the multi-view scenario

identify two different but valuable clusterings: characterizing the observations' color (first view: subspace $\{1, 2, 3\}$) and the observations' shape (second view: subspace $\{4, 5, 6\}$). If a single partitioning of this data in the full-space $\{1, \ldots, 6\}$ is enforced, the result will be very small, specialized clusters, e.g. 'blue rectangles'. For the purpose of generalization, 3 clusters in 2 views are preferable over 9 clusters considering all attributes. Especially for data with many attributes a mixture distribution in the full-space does not generalize the data well.

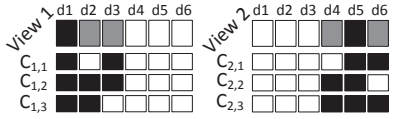
Taking a generative perspective, we can assume each object to be generated by *multiple mixture distributions*, each referring to a different view of the data. Consequentially, each object follows *multiple* components, each in a different mixture model, each defining a distribution only for a certain view (i.e., subspace) of the data, and each representing a different role of the object. This poses several challenges:

Challenge 1: Multiple Groupings. In the most simple scenario of multi-view data, the views do not share any characteristics (disjoint subspaces). A schematic representation of this case for a database with 6 dimensions, 2 views, each with 3 clusters, is given in Fig. 7.2(a). Intuitively, we can model this scenario by 'concatenating' several traditional mixture models. The question remains, how to appropriately model the relevant dimensions of each view.

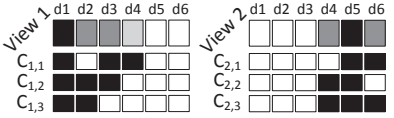
Challenge 2: Subspace Clusters. Usually, the solution via traditional mixture models, discussed above, is too restrictive for the characteristics of real world data. While for a certain view a set of attributes is relevant in general, we cannot expect that each cluster covers exactly the same set of dimensions as its view (subspace cluster w.r.t. the view). While the dimension 'viewers age' might be relevant for the view 'genre' in general (e.g., 'Horror' movies target only adults), some genres like, e.g., '3D Animations' show no certain characteristic in this dimension. This scenario corresponds to Fig. 7.1 and is illustrated in Fig. 7.2(b).



(a) Disjoint views



(b) Disjoint views & subspace clusters



(c) View overlaps & subspace clusters

Figure 7.2: Different scenarios of multi-view data (the shading represents the relevance of the dimensions; white: irrelevant dimensions; the darker the shade the more relevant the dimension)

Subspace clusters cannot be represented by traditional mixture models. While in the relevant dimensions of a subspace cluster, the attribute values are distributed according to, e.g., a Gaussian distribution, irrelevant dimensions follow a completely different model, e.g., a uniform distribution. That is, depending on the dimension's relevance a different parametric family is used. Thus, to model data containing subspace clusters, we encounter the challenge of model selection, i.e., before we can estimate the actual mixture model parameters, we first have to determine the parametric families that are used for each cluster.

Challenge 3: Overlapping Views. So far, we just discussed non-overlapping views. In general, however, dimensions can occur in multiple views (Fig. 7.2(c)). e.g., the dimensions 'gender' and 'age' might be characteristic for the two views 'hobby' and 'profession' of a customer database. This scenario is particularly challenging as several components might compete with each other for generating an object in one or more dimensions (e.g., clusters $C_{1,1}$ and $C_{2,3}$ in dimension four). Obviously, the *dominant* view (and, hence, dominant distribution) might vary for each dimension and each object: while in dimension 'gender' some objects rate the view 'hobby' as dominant, other objects use the view 'profession' in this dimension; in the dimension 'age', completely different views might be considered as dominant. This observation is even intensified by considering subspace

clusters: some clusters might not be relevant in the overlapping dimensions. Accordingly, it is not sufficient to consider only the views, but we need to consider the actual subspace clusters to determine the dominant view. Since each object might be located in different subspace clusters, different overlap scenarios can occur.

To tackle all these challenges, we propose a Bayesian framework modeling data with an inherent multi-view clustering structure. Our model:

- provides multiple generalizations of the data by modeling individual mixture models, each representing a distinct view
- handles individual sets of relevant dimensions for each cluster by performing Bayesian model selection
- tackles the ambiguity of the objects' memberships regarding multiple, competing components

7.2 Generative Multi-View Model

In this section, we introduce a Bayesian framework modeling the process of generating data containing multiple clustering views. An overview of our framework is given by the graphical model depicted in Fig. 7.3. We start in Section 7.2.1 by describing our model from a generative perspective, i.e., we show how our model generates data containing multiple views. The inverse process where a set of observations is *given* and the model's components are learned, is introduced in Section 7.2.2. Following convention, we do not distinguish between a random variable X and its realization $X = \mathbf{X}$ if it is clear from the context. As an abbreviation, we denote sets of random variables with the index $*$, e.g., $Y_{*,d}$ is the set of random variables $\{Y_{i,d}\}$ with i in the corresponding index domain, and Y is an abbreviation for the set $Y_{*,*}$.

7.2.1 Generating Multi-View Data

In our model, we explicitly differentiate between the relevant dimensions of the clusters and the relevant dimensions of the views. The relevant dimensions of a view provide a concise description for the relevant dimensions for a *set* of clusters. That is, the clusters belonging to the same view are located in similar subspace projections. Since the clusters' relevant dimensions might slightly vary, the relevance of dimensions for the view can also vary. In Figure 7.2(b), for example,

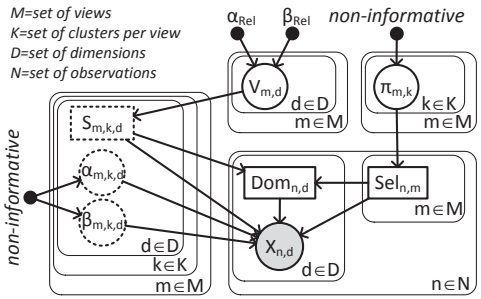


Figure 7.3: Graphical model of our method. Rectangles denote discrete random variables, circles continuous random variables, and black dots (deterministic) hyperparameters of the prior distributions.

dimension 1 has a high relevance for view 1 since all of its represented clusters use this dimension; dimension 1 is a good descriptor for the whole set of clusters. Dimension 3, in contrast, has a slightly smaller relevance since one of the clusters does not require this dimension. Thus, to reflect the differing relevances of dimensions $d \in D = \{1, \ldots, d_{max}\}$ for each view $m \in M = \{1, \ldots, m_{max}\}$, our model includes the (continuous) latent variables $V_{m,d}$ on $(0, 1)$.

Based on this relevance information, the actual relevant dimensions of each subspace cluster can be generated. We model this aspect by the (discrete) random variable $S_{m,k,d}$ on $\{0, 1\}$ for $d \in D$, $k \in K$ and $m \in M$.[1] The latent variable is 1 if the dimension d is relevant for the k-th cluster of view m, and 0 otherwise. The higher the relevance of a view's dimension, the more likely is the dimension relevant for the cluster. This property can be realized by a Bernoulli process. With probability $V_{m,d}$, the dimension d is relevant for the cluster, and with probability $1 - V_{m,d}$ it is not. Formally, the distribution of the latent variable $S_{m,k,d}$ is given by

$$\begin{aligned} p(S_{m,k,d} = 1 \mid V_{m,d} = r) &= r \\ p(S_{m,k,d} = 0 \mid V_{m,d} = r) &= (1 - r) \end{aligned} \tag{7.1}$$

If the value of $V_{m,d}$ is either close to 1 or close to 0, then the clusters in this view m are likely to have the same value for $S_{m,k,d}$. Thus, if the value of $V_{m,d}$ is either close to 1 or close to 0 for all dimensions $d \in D$, then the subspaces of the clusters in this view are very similar. If $V_{m,d}$ is close to 0.5, we do not have a clear presetting, and, thus, the subspaces of the clusters may differ stronger.

[1] To simplify our model description, we assume that each view $m \in M$ describes k_{max} clusters, i.e. $K = \{1, \ldots, k_{max}\}$.

Prior distributions. To allow a fully Bayesian approach, we specify prior distributions for the variables $V_{m,d}$. We select the prior according to a Beta distribution, i.e., $V_{m,d} \sim Beta(\alpha_{Rel}, \beta_{Rel})$ with hyperparameters $\alpha_{Rel} \in \mathbb{R}_{>0}$ and $\beta_{Rel} \in \mathbb{R}_{>0}$. A Beta distribution is suited due to the following reasons: First, since $V_{m,d}$ simulates a Bernoulli process, the Beta distribution corresponds to its conjugate prior. Second, based on the hyperparameters, the user can control the views' purity. That is, one can control the similarity between the clusters' subspaces originating from the same view. As mentioned above, high similarity between the subspaces can be realized by choosing $V_{m,d}$ close to 1 or close to 0. This issue can be modeled by selecting $\alpha_{Rel} = \beta_{Rel} < 1$. If no knowledge about the views' purity is given, we can simply choose $\alpha_{Rel} = \beta_{Rel} = 1$, leading to a non-informative prior.

Generating the membership information After generating the relevant dimensions of each cluster, we now aim at generating observations that follow multiple overlapping views. More precisely, in each of the views each object shall belong to a single cluster; thereby, we realize a single grouping within a single view and multiple overlapping groupings among different views.

This idea can be modeled by the latent variable $Sel_{n,m}$ on $K = \{1, \ldots, k_{max}\}$ that models which of the k_{max} clusters an object n follows in view m. The distribution of Sel is governed by the (relative) weights $\pi_{m,k}$ of the clusters, i.e.,

$$p(Sel_{n,m} = k \mid \pi_{m,*}) = \pi_{m,k}$$

As usual for mixture models, the larger the weight of a cluster, the more objects belong (in expectation) to the cluster. Please note that in contrast to traditional mixture models, in our model each view represents a certain grouping of *all* objects. Thus, we have $\sum_{k \in K} \pi_{m,k} = 1$ for each view $m \in M$, while in traditional mixture models the overall weight of all clusters is normalized to 1.

As discussed in challenge 3 (and illustrated in Fig. 7.2(c)), different views compete with each other. An object might belong to two clusters which both are marked as relevant in a specific dimension d. To solve the ambiguity about the object's membership in this dimension, we specify one of the views as dominant (for this object and dimension). This aspect is modeled by the latent variable $Dom_{n,d}$ on $M = \{1, \ldots, m_{max}\}$. Here, a view $m \in M$ can only be dominant in d if the selected cluster is also relevant in d. Thus, let $M'_{n,d} = \{m' \in M \mid S_{m', Sel_{n,m'}, d} = 1\}$ be the set of views that are potentially dominant for object n in dimension d, the distribution of $Dom_{n,d}$ is modeled by

$$p(Dom_{n,d} = m \mid Sel_{n,*}, S_{*,*,d}) = \begin{cases} 1/|M'_{n,d}| & \text{if } m \in M'_{n,d} \\ 0 & \text{if } m \notin M'_{n,d} \wedge M'_{n,d} \neq \emptyset \\ 1/|M| & \text{else} \end{cases}$$

That is, we randomly select a view from the potentially dominant views (case 1), while the remaining views cannot be selected (case 2). The third case just occurs, if none of the selected clusters of an object is relevant in this dimension. In this case, an arbitrary view can be selected as dominant since any cluster represents just noise in this dimension.

Generating Observations Finally, we specify the distributions from which the attribute values of a cluster are sampled, i.e., we model the actual components of the multiple mixture models. However, keep in mind that for subspace clustering we have two different parametric families: $\mathcal{K}_1$ for the relevant dimensions and $\mathcal{K}_0$ for irrelevant ones.

In our model, we select the parametric family $\mathcal{K}_1$ according to the set of Beta distributions, i.e., we consider a mixture of Beta distributions. This is advantageous compared to the frequently used Gaussian distributions since Gaussian distributions have an infinite support, which usually does not match the observed data. In many applications, we have a finite attribute domain that can be normalized to the range $(0, 1)$; this is exactly captured by the Beta distribution (cf. Fig. 7.4). Additionally, the Beta distribution is able to model distributions near the border of the data space. These Beta distributions are modeled by the two random variables $\alpha_{m,k,d}$ and $\beta_{m,k,d}$ on $\mathbb{R}_{>0}$ providing the necessary shape parameters of each distribution (for each view m, each cluster k, and each dimension d). For the parametric family $\mathcal{K}_0$, we simply use the uniform distribution on $(0, 1)$ since this corresponds to a noisy dimension. Thus, $|\mathcal{K}_0| = 1$ holds.

Which parametric family a mixture component in dimension d belongs to was modeled by the latent variable S. Thus, finally, the attribute values of each object can be modeled by the random variable $X_{n,d}$ with distribution:

$$X_{n,d} \mid Dom_{n,d}, Sel_{n,*}, S_{*,*,d}, \alpha_{*,*,d}, \beta_{*,*,d} \sim \begin{cases} Beta(\alpha_{i,j,d}, \beta_{i,j,d}) & \text{if } S_{i,j,d} = 1 \\ Uni(0, 1) & \text{else} \end{cases}$$

where $Dom_{n,d}{=}i$ and $Sel_{n,i}{=}j$. Thus, for each dimension d, an object follows the distribution given by the *selected* cluster in the *dominant* view.

$$Beta(x;\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \cdot x^{\alpha-1} \cdot (1-x)^{\beta-1}$$

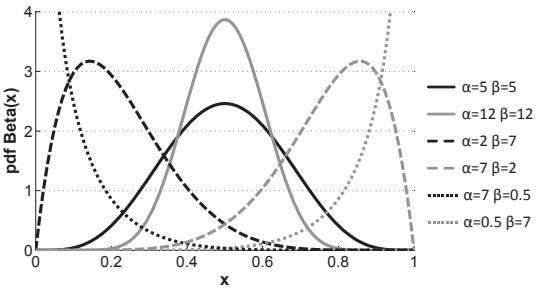


Figure 7.4: Probablity density function of the Beta distribution

Prior distributions. Again, we choose appropriate prior distributions to enable inference. We select non-informative priors since usually no further knowledge is provided about the data's clustering structure. A non-informative prior for the cluster weights $\pi_{m,k}$ is simply realized by choosing $p(\pi_{m,1},\ldots,\pi_{m,k_{max}}) = const$ for each view m.

For the variables $\alpha_{m,k,d}$ and $\beta_{m,k,d}$, we suggest a non-informative prior $p(\alpha,\beta)$ that ensures a uniform distribution over the mean and variance of the resulting Beta distributions $Beta(\alpha,\beta)$. Intuitively, this way the cluster centers are uniformly selected from the domain $(0,1)$ and the variance from the domain $(0,\frac{1}{12})$. Thus, the prior fulfills

$$\int_\alpha \int_\beta p(\alpha,\beta) \cdot \mathbf{1}(\mathbb{E}(Beta(\alpha,\beta)) = x)\ \mathrm{d}\alpha\mathrm{d}\beta \sim Uni(0,1)$$

regarding the mean x of the resulting Beta distribution (same for the variance with $Uni(0,\frac{1}{12})$). We can approximately[2] achieve these properties by selecting the priors according to exponential distributions with rate parameter 0.1, i.e.,

$$\alpha_{m,k,d} \sim Exp(0.1)\ \ ,\ \ \beta_{m,k,d} \sim Exp(0.1)$$

7.2.2 Learning Objective

In the following, we describe our learning objective if a set of observed data points $\mathbf{X}$ is given. Usually, the learning objective would be to maximize the a posteriori probability $p(V,S,\alpha,\beta,Dom,Sel,\pi \mid X = \mathbf{X})$.

[2]Indeed, the distribution of the mean is exactly captured since the Beta distribution's mean is given by $\frac{\alpha}{\alpha+\beta}$, and for any λ it holds: $X,Y \sim Exp(\lambda) \Rightarrow \frac{X}{X+Y} \sim Uni(0,1)$

For our model, however, this idea is not meaningful since in this case usually all dimensions of a cluster are relevant: the data's likelihood is always higher when selecting a (certain) Beta distribution in contrast to selecting a uniform distribution. This is obvious since a uniform distribution is a special case of a Beta distribution with shape parameters $\alpha = \beta = 1$ and, hence, $\mathcal{K}_0 \subset \mathcal{K}_1$. Thus, simply determining the maximum a posteriori (MAP) estimate as given above leads to the problem of overfitting since a complex model obviously fits the data better than a simple one[3]; one would only choose relevant dimensions.

To overcome this problem, we first perform a model selection before learning the subspaces S and the shape parameters of the Beta distributions. That is, we balance the models' goodness of fit and their simplicity[4]. Thus, our learning objective is separated in two phases:

First, we perform Bayesian model selection [Bis06] by finding the best realization for V, Dom, Sel, and π. That is, we determine the MAP estimate

$$(\mathbf{V}^*, \mathbf{Dom}^*, \mathbf{Sel}^*, \boldsymbol{\pi}^*) = \underset{(\mathbf{V},\mathbf{Dom},\mathbf{Sel},\boldsymbol{\pi})}{\arg\max}\ p(V=\mathbf{V}, Dom=\mathbf{Dom}, Sel=\mathbf{Sel}, \pi=\boldsymbol{\pi} \mid X=\mathbf{X})$$

These variables are illustrated in our graphical model with solid lines. Since learning these variables involves a marginalization over S, α, and β, we realize the balancing of the model's complexity and its goodness of fit. Thus, due to this model selection step, some dimensions might be irrelevant for certain views, corresponding to a more simple model.

Since after the first phase the cluster model is determined, we can estimate in the second phase the actual mixture components and the clusters' subspaces. That is, we can now determine the MAP estimate for the variables S, α and β:

$$(\mathbf{S}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \underset{(\mathbf{S},\boldsymbol{\alpha},\boldsymbol{\beta})}{\arg\max}\, p(S=\mathbf{S}, \alpha=\boldsymbol{\alpha}, \beta=\boldsymbol{\beta} \mid X=\mathbf{X}, V=\mathbf{V}^*, Dom=\mathbf{Dom}^*, Sel=\mathbf{Sel}^*, \pi=\boldsymbol{\pi}^*)$$

Overall, our model allows to learn the clustering structure of data containing multiple overlapping views by using multiple mixture models. Clusters, i.e., mixture components, are located in individual subspace projections and are summarized by views through a concise description of their relevant dimensions.

[3] A similar example is polynomial interpolation: since the set of polynomials with degree x is a subset of the ones with degree $x+1$, the interpolation error decreases with increasing degree.

[4] In the example of polynomial interpolation, one balances the degree of the polynomial against its regression error.

7.3 The MVGen Algorithm

In this section, we introduce our MVGen (Multi-View Generative Model) algorithm that learns the multi-view clustering structure given a set of observed data points. Since exactly computing the MAP estimate $p(V, Dom, Sel, \pi \mid X)$ is intractable, we compute approximations that can be efficiently determined. In general, we exploit the principle of iterated conditional modes (ICM [Bes86]), which can be regarded as a greedy variant of the Gibbs sampling method [Bis06]. Instead of considering a complex joint distribution $p(A_1, \ldots, A_n)$, we iteratively maximize a set of conditional probabilities $p(A_i \mid A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n)$ until the process converges. This way, the random variables A_i are updated sequentially. The traditional k-means processing scheme can be seen as an instance of the ICM principle with just two easily computable update steps: recomputation of means and reassignment of points to clusters.

7.3.1 Update Equations

We briefly present the update equations required in our algorithm; they are summarized in Equation (7.U1)-(7.U4).

Updating the views V We start with the variable $V_{m,d}$, i.e., for each $m \in M, d \in D$ we aim at maximizing

$$p(V_{m,d} \mid V \backslash \{V_{m,d}\}, Dom, Sel, \pi, X) \propto \sum_S \int_\alpha \int_\beta p(V, Dom, Sel, \pi, S, \alpha, \beta, X) \, \mathrm{d}\alpha \mathrm{d}\beta \quad (7.2)$$

The most important aspect here is that we have to marginalize over the variable S, α, and β, as stated in Section 7.2.2. Only if the model selection step is performed, we can estimate α, β, and S.

In the appendix on page VII, we show the detailed derivation for the following Equation 7.U1 of the optimal realization of $V_{m,d}$:

$$V_{m,d} = \arg \max_{x \in (0,1)} c_a \cdot \log x + c_b \cdot \log(1 - x) + c_c \cdot \log(x + c_d) + \sum_{k \in K} \log(c_k \cdot x + 1) \quad (7.U1)$$

where the c_* are constant values given by

$$c_a = \alpha_{Rel} - 1 + |N_{m,d}| \qquad c_b = \beta_{Rel} - 1 \qquad c_c = -|N|$$

$$c_d = \textstyle\sum_{m' \in M, m' \neq m} V_{m',d} \qquad c_k = Beta_d(N_{m,k,d}) - 1$$

Here, $N_{m,d} = \{n \in N \mid Dom_{n,d} = m\}$ is the set of all observations that choose the view m in dimension d as *dominant*, and $N_{m,k,d} = \{n \in N_{m,d} \mid Sel_{n,m} = k\}$ are those observations which additionally *select* the cluster k. The term $Beta_d(N_{m,k,d})$ is computed based on the following equation

$$Beta_d(I) := \Big[\prod_{n \in I} Beta(X_{n,d}; \alpha_{MAP}, \beta_{MAP})\Big] \;/\; |I| \tag{7.3}$$

where I is an index set denoting which observations are considered, and α_{MAP} and β_{MAP} are the MAP estimates of the Beta distribution's shape parameters using the set I of observations. The function $Beta_d(I)$ approximates the term

$$\int_\alpha \int_\beta p(\alpha)p(\beta) \prod_{n \in I} Beta(X_{n,d}; \alpha, \beta)\, \mathrm{d}\alpha\mathrm{d}\beta$$

which has to be solved during the derivation of our update equations. $Beta_d(I)$ exploits the Bayesian Information Criterion (BIC, or Schwarz criterion [Sch78, Bis06]) in combination with the observation that in our case the Beta distribution is controlled by two free parameters.

Overall, Equation 7.U1 describes a simple univariate function in the variable x whose optimization can, for example, be done by Brent's algorithm [Bre73].

Updating π We perform a block update for the variables $\pi_{m,*}$. Since we use a non-informative prior, maximizing
$p(\pi_{m,*} \mid V, Dom, Sel, \pi \backslash \{\pi_{m,*}\}, X)$ is simply obtained by

$$\pi_{m,k} = |\{n \in N \mid Sel_{n,m} = k\}| \cdot |N|^{-1} \quad \forall k \in K \tag{7.U2}$$

Updating Dom **and** Sel Finally, we derive the update equations for the variables Dom and Sel. We perform a block update of the variables $Sel_{n,m}$ and $Dom_{n,*}$, i.e., for each observation n, we simultaneously update its selected cluster in view m and its dominant views over all dimensions. Formally, we aim at maximizing: $p(Sel_{n,m}, Dom_{n,*} \mid V, Dom \backslash \{Dom_{n,*}\}, Sel \backslash \{Sel_{n,m}\}, \pi, X)$

$$\propto \sum_S \int_\alpha \int_\beta p(S \mid V) p(Sel_{n,m} \mid \pi) \prod_{d \in D} \Big[p(Dom_{n,d} \mid S, Sel) \\ p(\alpha)p(\beta) \prod_{n' \in N} p(X_{n',d} \mid Dom, Sel, S, \alpha, \beta) \Big] \, \mathrm{d}\alpha\mathrm{d}\beta \quad (7.4)$$

We first resolve the integral over α and β by again using the BIC approximation (cf. Eq. 7.3). We assume that the MAP estimates α^{MAP} and β^{MAP} derivable from the current grouping change only marginally when reassigning a single point n to a different cluster. Similarly, the cluster sizes change only marginally, i.e., the sets $N_{m,k,d}$ differ by at most one element when reassigning observation n to a different cluster. By using this idea, we can substitute the part $p(\alpha)p(\beta)\prod_{n' \in N} p(X_{n',d} \mid Dom, Sel, S, \alpha, \beta)$ by $Beta(X_{n,d}; \alpha^{MAP}_{i,j,d}, \beta^{MAP}_{i,j,d})$ if $S_{i,j,d} = 1$ and by 1 (uniform distribution) if $S_{i,j,d} = 0$.[5] This simplification stems from the fact that with given, constant MAP estimates also the densities $p(X_{n',d} \mid ...)$ for $n' \neq n$ are constant. Thus, Eq. 7.4 simplifies to:

$$\propto \sum_S p(S \mid V) p(Sel_{n,m} \mid \pi) \prod_{d \in D} \Big[p(Dom_{n,d} \mid S, Sel) \\ \begin{cases} Beta(X_{n,d}; \alpha^{MAP}_{i,j,d}, \beta^{MAP}_{i,j,d}) & \text{if } S_{i,j,d} = 1 \\ 1 & else \end{cases} \Big] \quad (7.5)$$

Due to the integration over all possible realizations of S, the term $p(Dom_{n,d} \mid S, Sel)$ can well be approximated by the expected dominance of a view m in dimension d. The expected dominance is given by

$$ED_d(m) := \frac{V_{m,d}}{\sum_{m' \in M} V_{m',d}}$$

Thus, the above equation simplifies to

$$\propto \sum_S p(S \mid V) p(Sel_{n,m} \mid \pi) \prod_{d \in D} \Big[ED_d(Dom_{n,d}) \\ \begin{cases} Beta(X_{n,d}; \alpha^{MAP}_{i,j,d}, \beta^{MAP}_{i,j,d}) & \text{if } S_{i,j,d} = 1 \\ 1 & else \end{cases} \Big] \quad (7.6)$$

Since the $S_{m,k,d}$ are independent given V, the summation over S is effective only for the variable $S_{i,j,d}$. Thus, the summation vanishes when making the two cases of $S_{i,j,d}$ explicit. That is, we introduce the term

$$AD_d(m,k) := p(S_{m,k,d} = 1 \mid V) \cdot Beta(X_{n,d}; \alpha^{MAP}_{m,k,d}, \beta^{MAP}_{m,k,d}) + p(S_{m,k,d} = 0 \mid V) \cdot 1$$

[5] We use the abbreviations $i{:=}Dom_{n,d}$ and $j{:=}Sel_{n,Dom_{n,d}}$

and Equation 7.6 simplifies to

$$p(Sel_{n,m} \mid \pi) \cdot \prod_{d \in D} ED_d(Dom_{n,d}) AD_d(Dom_{n,d}, Sel_{n,Dom_{n,d}})$$

Please note that the functions ED_d and AD_d are independent of Sel and Dom and fully specified by the values of V, α^{MAP}, and β^{MAP} (which are given!). Thus, while updating the values of Dom and Sel, we do *not* have to recompute the functions ED_d and AD_d.

Based on the above equation, it becomes apparent that $Dom_{n,*}$ can be optimized for each dimension individually. Especially, if the variable Sel is *given*, we can efficiently compute the optimal realization of $Dom_{n,d}$ by

$$Dom_{n,d} = \arg\max_{m \in M} ED_d(m) AD_d(m, Sel_{n,m}) \tag{7.U3}$$

As shown, the optimal realization of Dom depends on Sel in a simple way. Thus, we can focus on finding a good solution for $Sel_{n,m}$. The optimal solution of $Sel_{n,m}$ can efficiently be computed by

$$Sel_{n,m} = \arg\max_{k \in K} \left\{ \pi_{m,k} \prod_{d \in D} \max\left\{ ED_d(m) \cdot AD_d(m,k), c_{m,d} \right\} \right\} \tag{7.U4}$$

where $c_{m,d} = \max_{m' \in M, m' \neq m} ED_d(m') \cdot AD_d(m', Sel_{n,m'})$. The value of $c_{m,d}$ is constant since it neither depends on $Sel_{n,m}$ nor on $Dom_{n,*}$. Please note that the update of $Sel_{n,m}$ directly uses the best solution for $Dom_{n,*}$. Thus, we do not have to optimize $Dom_{n,*}$ separately but the optimal values are computed while updating $Sel_{n,m}$.

7.3.2 Recommended Update Sequence

Given the derived update equations, any sequence that recurrently invokes each of these equations is possible to determine a desired clustering solutions. However, based on the dependencies as given by our graphical model and the particular role of the model selection phase, we recommend the following update sequence for the random variables:

1. We sequentially update the variables $V_{m,d}$ for each $m \in M, d \in D$ until the views are stable. [Eq. (7.U1)]

2. For each object $n \in N$, we sequentially update the variables $Sel_{n,*}$ and $Dom_{n,*}$ until Sel and Dom are stable.
   (a) To update $Sel_{n,*}$ and $Dom_{n,*}$, we sequentially update $Sel_{n,m}$ for each $m \in M$ until $Sel_{n,*}$ is stable. [Eq. (7.U3) & (7.U4)]
3. Update of $\pi_{m,*}$ for each $m \in M$ [Eq. (7.U2)]; goto step (2) until the process has converged.
4. goto step (1) until the process has converged.

Thus, overall, we exploit the ICM principle in a nested fashion. The outer loop iterating over steps 1 and 2/3 represents the alternation between learning the views and learning the groupings. For the inner loop, iterating over 2 and 3, the views are given and we try to optimize the cluster assignments as good as possible. Note that implicitly also the mixture components α and β are optimized since based on the BIC approximation their MAP estimates are considered.

Initialization To complete the above algorithm, we describe a straightforward initialization. We simply initialize Sel by the following method: For each dimension $d \in D$, we apply the k-means method with $k = |K|$. Thus, leading to $|D|$ many clusterings. Since, however, Sel requires just $|M|$ different views, we follow an approach inspired by traditional agglomerative clustering methods: To reduce the number of clusterings, we successively determine those clusterings that are most similar to each other, and we merge these clusterings to a single one. Thus, in each step the number of clusterings is reduced by one until the required number $|M|$ is reached. To merge two clusterings, we simply union the corresponding sets of dimensions and we recompute the k-means result in the novel space. As similarity measure between the clusterings, we use the F-measure [WXC09].

After initializing Sel, the variable π is determined based on Equation 7.U2. Since no information about the views is given, Dom is initialized randomly. The variable V is also initialized randomly based on its prior distribution.

7.3.3 Determining Components and Subspaces

According to Section 7.2.2, the second phase of our learning objective is to determine the MAP estimate of $p(S, \alpha, \beta \mid X, V, Dom, Sel, \pi)$.

Since the variables Dom and Sel are given, the set of observations that contribute to the Beta distribution of cluster k in dimension d and view m is known, and was denoted by the set $N_{m,k,d} = \{n \in N_{m,d} \mid Sel_{n,m} = k\}$. Thus, the shape

parameters $\alpha_{m,k,d}$ and $\beta_{m,k,d}$ of each mixture component simply correspond to their MAP estimate given the set of observations $N_{m,k,d}$.

In general, however, determining the MAP estimate for the shape parameters of a Beta distribution is not possible in closed form; one has to iteratively solve systems of equations [BT78]. Since this is highly inefficient, we refer to the commonly used approach of moment matching: the shape parameters are computed based on the mean and variance values of the observations. This approach is in line with the non-informative prior distributions of α and β, which do not favor certain means or variance values. Thus, we get:

$$\alpha_{m,k,d} = \frac{(1-\mu)\mu^2 - \mu \cdot \sigma^2}{\sigma^2} \text{ and } \beta_{m,k,d} = \frac{\mu(\mu-1)^2}{\sigma^2} + \mu - 1$$

where μ denotes the mean of the observations $\{X_{n,d} \mid n \in N_{m,k,d}\}$, and σ^2 the variance, respectively. Note that these equations are also used for the MAP estimates required in Section 7.3.1. Thus, the estimates for α^{MAP} and β^{MAP} can be efficiently computed in these steps.

Finally, an estimate for the random variables S can be obtained by testing which model – relevant or irrelevant dimension – is more likely. That is, if

$$p(S_{m,k,d} = 1 \mid V) \cdot \prod_{n \in N_{m,k,d}} Beta(X_{n,d}; \alpha_{m,k,d}, \beta_{m,k,d}) > p(S_{m,k,d} = 0 \mid V) \cdot 1$$

, the dimension d of cluster k in view m, will be relevant. Here, we do not have to refer to the BIC approximation but can use the likelihood of the Beta distribution. Since the view V is already learned, i.e., the model selection is done, the trade-off between the models' complexities and their goodness of fit is already reflected in the term $p(S_{m,k,d} \mid V)$.

7.4 Related Work

Our MVGen exploits a Bayesian framework to model the generation of data with an underlying multi-view clustering structure. We discuss three paradigms related to this topic:

Subspace clustering: In contrast to traditional full-space clustering, subspace clustering (co-clustering/bi-clustering) [KKZ09] assumes that for each cluster an individual subset of attributes might be irrelevant. These locally irrelevant attributes cause an obfuscation of the clustering structure in the full-space, which makes full-space approaches futile. The consideration of attribute subsets is

highly related to our multi-view scenario, where different views of the data are most likely reflected by different attributes. However, subspace clustering does neither realize a grouping of clusters to reflect partitionings under several views nor is it aware of the varying competition and dominance of multiple clusters concerning the attribute values of the data. Therefore, it does not meet the requirements for multi-view clustering.

Multi-view clustering: The paradigm of multi-view or alternative clustering meets our goal of revealing the cluster structure of multi-faceted data. Three different categories are identified in [MGFS10]. The first category's representatives, e.g., [JMD08, BB06, DB10a], operate in the full-space and, therefore, suffer from similar problems as traditional clustering. Furthermore, they are usually focused on determining just two alternative clusterings, whereas for complex datasets multiple views can be expected. Approaches from the second category detect clusters in subspace projections ([NDJ10], OSCLU, ASCLU). However, [NDJ10] cannot handle overlapping views and does not allow individual subspaces per clusters, and our OSCLU and ASCLU approaches (cf. Chapters 4 & 5) do not provide a grouping into views, i.e., the views remain unknown. Approaches of the last category iteratively determine an alternative clustering based on the previous one via space transformations such as PCA ([CFD07, DB13b]) or distortion of the distance function ([DQ08, QD09]). Distortions of the original space like this, usually hinder an intuitive interpretation of the clustering result. Contrarily, axis-parallel projections of the data, as for our approach, directly refers to the originating attributes for each cluster.

Model-based clustering: This general paradigm assumes the considered data to be sampled from a statistical model. Several approaches for estimating the parameters of the underlying probability distributions, e.g., to maximize the log-likelihood of the data, were proposed including the EM algorithm and ICM [Bis06, Bes86]. Model-based clustering is very flexible as the modeled distributions can be arbitrarily complex. Traditionally, such approaches use a *single* mixture distribution (which spans across all dimensions of the data space). Even though each observation might be associated with a membership degree (e.g., the likelihood of belonging to a cluster), such a principle of *soft clustering* does not support the idea of generating objects through multiple components as for the multi-view scenario. To overcome this issue, a few models ([SJR10, FB08, BKG$^+$05]) try to represent such multi-component membership (i.e., overlapping clusters). However, they are still not suited for multi-view clus-

tering: They do not consider a grouping of clusters into views, i.e., they do not model that an object takes a single role within a single view but different roles among different views. Instead, these models lead to results where an object might take multiple roles within a single view. Note that global dimensionality reduction and feature selection [PZCW10] also do not solve our task: First, we consider multiple views in multiple different subspaces. Second, in our model each cluster is associated with an individual subspace projection.

Overall, none of the existing methods is able to handle multiple views that compete against each other in overlapping dimensions and containing clusters with individual sets of relevant dimensions. Our novel statistical model handles all these aspects.

7.5 Experimental Analysis

Setup We compare MVGen with the multi-view clustering techniques *Multi-View 1* and *Multi-View 2* proposed in [CFD07] and with two variants of the *Alternative Clustering* method proposed in [QD09]. These approaches best reflect the demands for multi-view clustering as discussed in Sec. 7.4. Additionally, we use two variants of the k-means method.

For case studies on real world data we use the CMUFaces, liver disorders, diabetes, iris and vowel data (all from the UCI repository [FA10]), and Escher images. Synthetic data containing multiple views is generated based on our generative model. The default dataset contains 2 disjoint views, each with 5 clusters, 10 dimensions, 1000 objects, and the clusters' subspaces deviate to the views' dimensions by 5%.

The methods of [CFD07] and MVGen are provided with the number m_{max} of views and the number k_{max} of clusters per view. Since [QD09] just detects two groupings, we use two variants: 1) The true number of clusters per view is set. In this case the method detects $2 \cdot k_{max}$ clusters. 2) We parametrize the method with $k_{max} \cdot m_{max}/2$. In this case, $k_{max} \cdot m_{max}$ clusters are detected, which matches the overall number of clusters hidden in the data. Similarly, we use for k-means $k = k_{max}$ as well as $k = k_{max} \cdot m_{max}$. Runtime is measured on 2.33GHz Intel XEON CPU with 8 GB main memory. Quality is assessed based on the $E4SC$ measure (cf. Chapter 13) used in evaluation of subspace clustering. Since MVGen is the only method performing multi-view *subspace* clustering, we do not evaluate the subspaces of the competing methods but just concentrate on their detected object

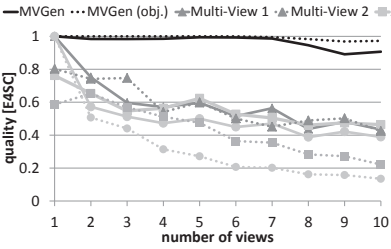


Figure 7.5: Varying number of views

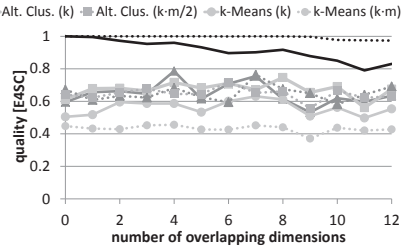


Figure 7.6: Effects of overlapping views

groupings. To enable a direct comparison of MVGen with these approaches, we also include the results of MVGen if we ignore the subspaces in its evaluation, denoted with 'MVGen (obj.)'.

7.5.1 Evaluation on Synthetic Data

In Fig. 7.5, we vary the number of hidden views in the data. The overall dimensionality of the data is 30. As depicted, MVGen is the only approach able to detect the clustering structure in the case of many views. The clustering quality is very high, even if we incorporate the detected subspaces in our evaluation (solid line of MVGen). Obviously, the quality is even higher if we evaluate the object groupings only (dashed line). The competing methods behave differently: while for single-view data the quality is relatively high, their quality heavily decreases with an increasing number of views. Interestingly, for a high number of views, the quality of the two multi-view techniques (depicted by triangles) is not much larger than the one of the k-means method with $k = k_{max}$. These methods are not well suited to analyze data containing multiple views.

Next, we analyze the potential of our method to detect overlapping views. In Fig. 7.6, we use a dataset with 12 dimensions containing 3 views. We vary the number of overlapping dimensions, i.e., dimensions that occur in more than one view, until each dimension occurs in two views. As shown, the methods are nearly not influenced by overlapping dimensions. The reason might be that none of the views is completely contained in another one. One is still able to detect the clusters of each view. MVGen detects the object groupings almost perfectly; some of the clusters' relevant dimensions are missed for high overlapping degrees. Note: The good quality of the competing methods is only observed because we just have 3 views in this experiment.

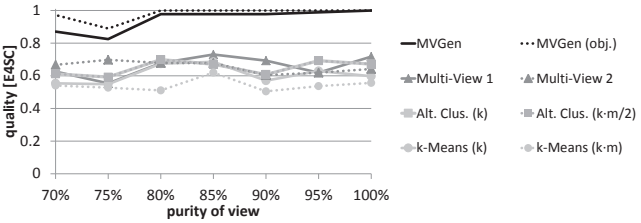


Figure 7.7: Effects of the views' purity

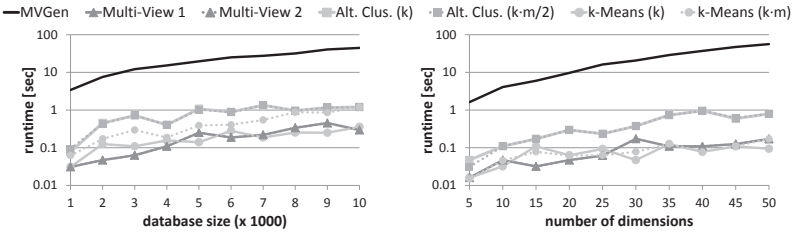


Figure 7.8: Runtime vs. database size

Figure 7.9: Runtime vs. dimensionality

In Fig. 7.7, we show that MVGen is able to find subspace clusters (located in subspace views). In our model, we allow a certain deviation of the clusters' subspaces to the relevant dimensions of their views; here, denoted as the purity. In this experiment, we vary the purity from 70% to 100%. Since each view covers 5 dimensions, a purity of 70% leads to subspace clusters covering now only 3 relevant dimensions. As shown, MVGen succesfully detects the relevant dimensions of each cluster. Since we use a model selection approach, we trade-off the simplicity of the model against its goodness of fit. For the competing method, no conclusion can be drawn since for their evalution, we do not consider subspaces.

Scalability Even though our focus is on clustering quality, we briefly analyze MVGen's efficiency. In Fig. 7.8, we increase the number of objects in the database. All methods show increasing runtime and the slopes of the curves are in a similar range. Please note that the two approaches Multi-View 1 & 2 have almost identical runtimes, and, since we use 2 views, the two Alt. Clus. approaches are also identical in their runtimes. Apparently, the absolute runtime of our method is the highest due to the complex model selection phase that trades off relevant and irrelevant dimensions. However, the absolute runtime of MVGen is still low. Fur-

thermore, as we believe, the higher runtime is compensated by the significantly higher clustering quality of MVGen. In Fig. 7.9, we increase the dimensionality of the dataset. We observe a similar behavior as in the previous experiment.

Overall, MVGen shows good scalability and it is the only method simultaneously achieving high clustering qualities.

7.5.2 Evaluation on Real World Data

For evaluation on real world data, we use different evaluation principles, all focusing on the aspect of detecting multiple views. In our first experiment, we extend the datasets iris and vowel to data containing multiple views: for this, we randomly concatenate the attribute values of different objects to a higher dimensional space. The original datasets have dimensionalities of 4 and 10, respectively, while the extension to multi-view data leads to dimensionalities up to $9 \cdot 4 = 36$ (iris) and $6 \cdot 10 = 60$ (vowel), respectively. Figures 7.10 & 7.11 show the results: For a small number of views, the quality of some competing approaches is similar to the one of MVGen. However, increasing the number of views leads to a decreasing clustering quality for all competing approaches. In contrast, MVGen shows constant quality values; MVGen is not affected by an increasing number of views but detects the different object groupings even for a high number of views. These results for real world data are consistent with the observations made for the synthetic data.

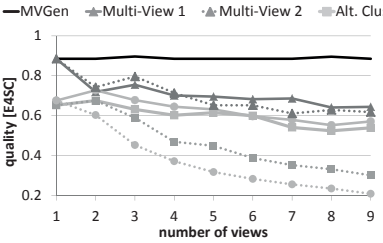


Figure 7.10: Quality on iris data

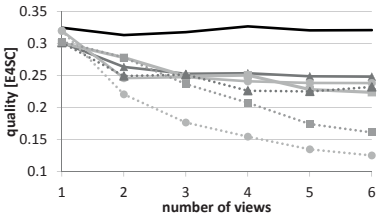


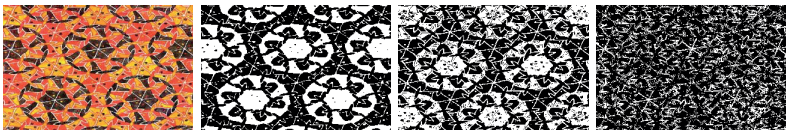
Figure 7.11: Quality on vowel data

In the next experiment, we analyze the clustering result of MVGen on the CMUFace data. This data is interesting for multi-view clustering since it consists of images taken from persons showing varying characteristics as their facial expressions (neutral, happy, sad, angry), head positions (left, right, straight), and

eye states (open, sunglasses). As also done in [DB10b], we randomly select 3 persons with all their images and applied PCA retaining at least 90% of the data's variance as a pre-processing. The result of MVGen for two views each with three clusters is illustrated in Fig. 7.12. The images correspond to the means of each detected cluster. By visual inspection, we can easily find the reason for detecting these two views: The first view, describes the grouping based on the different persons, while the second view, corresponds to a grouping based on their head positions.

Figure 7.12: Result of MVGen on face data

Next, we perform an experiment as introduced in [QD09]. They propose to perform image segmentation on Escher images, which are known to have multiple interpretations to the human eye. For clustering, each pixel is regarded as an object with RGB and HSV values as features. In Fig. 7.13 (left), such an image is depicted (followed by the three views detected by MVGen). Focusing on the dark regions, there is a segmentation of the image as given by the first view of MVGen. This segmentation is dominant since the dark parts clearly deviate from the orange/yellow parts. However, MVGen is also able to discover the more subtle view where the yellow parts are decoupled from the others. Most interesting is the third view detected by MVGen: it corresponds to only the background of the image. For the other methods we observed the following: The work of [QD09] was only able to detect groupings similar to MVGen's first and second view (as also shown in [QD09]). Interestingly, the work of [CFD07], which is designed to detect more than 2 views, was only able to find view 1. The detected 'alternative' groupings were all similar. None of the competing methods was able to detect the third, background view.



Original image View 1 View 2 View 3

Figure 7.13: Result of MVGen on an Escher image

In our last case study, we want to highlight the benefit of explicitly modeling the relevant subspace for each view, as done by MVGen. Knowing the relevant attributes, enables us to reason about the views' context and to explain the clusters. Table 7.1 depicts for the liver disorders and diabetes data the detected subspaces of each view. The number of clusters per view was chosen as 2. As shown for the liver disorders data, the two views/clusterings clearly differ from each other (small rand index), and the views do not correspond to the full-dimensional space, i.e., for each view some dimensions are irrelevant. For liver, we observe disjoint views. The first view clearly describes the relation between alcohol consumption and the mean corpuscular volume, while the second view represents the weaker indicators. On the diabetes data, the detected views match well to some factors causing diabetes of type 1 or type 2 (adult-onset diabetes; also caused by high blood glucose levels during pregnancy). Here, a further interesting observation can be made: Besides finding dissimilar groupings in subspace projections, we now also get slightly overlapping views. For example, the dimension 'body mass index' is relevant for both clusters in view 1 and for a single cluster in view 2. This result also confirms our hypothesis that the clusters of the *same* view may slightly differ in their relevant dimensions.

Liver Disorders Data (rand index between views: 0.25)	
V1	mean corpuscular volume, number of half-pint equivalents of alcoholic beverages drunk per day
V2	alkaline phosphotase, alamine aminotransferase, aspartate aminotransferase, gamma-glutamyl transpeptidase
Diabetes Data (rand index between views: 0.51)	
V1	body mass index, diabetes pedigree function, triceps skin fold thickness, 2-hour serum insulin; (for one cluster: plasma glucose concentration)
V2	age, diastolic blood pressure, # of times pregnant, plasma glucose concentration; (for one cluster: body mass index)

Table 7.1: Subspace views on liver and diabetes

Overall, our experiments show that MVGen successfully detects the multi-view clustering structure on a variety of real world datasets.

7.6 Conclusion

Our MVGen approach successfully exploits the model-based clustering paradigm for the multi-view context. Our Bayesian framework accounts for the challenges of multiple, overlapping, and competing mixture distributions for differing views. Since each view reflects specific characteristics of the data, each mixture component is defined in an individual subspace. The comparison of MVGen with competing approaches demonstrated the strengths of detecting views in multiple subspace projections. Our MVGen approach was able to discover multiple clustering views for various real world datasets. Especially the explicit modeling of the views' relevant subspaces has proven to be very valuable for interpreting the final clustering results.

8

Semi-Supervised Multi-View Clustering in Subspace Projections

OFTEN users are able to provide partial prior information regarding the clustering structure. Semi-supervised clustering techniques have shown to substantially improve clustering results for *single-view clustering* by integrating such prior knowledge into the clustering process. In this chapter, we want to present an approach to join the research areas of multi-view and semi-supervised clustering to integrate prior knowledge in the process of detecting multiple clusterings.

We propose a Bayesian framework modeling multiple clusterings of the data by multiple mixture distributions, each responsible for an individual set of relevant dimensions. In addition, our model is able to handle prior knowledge in the form of instance-level constraints indicating which objects should or should not be grouped together. Since a priori the assignment of constraints to specific views is not necessarily known, our technique automatically determines their membership. For efficient learning, we propose the algorithm SMVC using variational Bayesian methods. With experiments on various real-world data, we demonstrate SMVC's potential to detect multiple clustering views and its capability to improve the result by exploiting prior knowledge.

8.1 Introduction

Semi-supervised clustering techniques [BDW08] try to incorporate the user's preferences by exploiting prior knowledge during the clustering process. For traditional single-view clustering, these techniques have shown to substantially increase the clustering results. Motivated by the success of both research areas, multi-view clustering and semi-supervised clustering, we propose a semi-supervised multi-view clustering technique. Our goal is to exploit user provided prior knowledge to enhance the results of multiple, alternative clusterings.

For semi-supervised clustering, it is crucial that the user can provide supervision in an easy and understandable way. While cluster level constraints, such as the clusters' sizes, positions, or distributions, usually require an abstract understanding of the desired clustering structure, instance level constraints, which, e.g., indicate partial information about cluster memberships, are much more intuitive. A popular way of modeling such prior information is via equivalence constraints, which indicate for pairs of instances whether they should belong to the same cluster (must-link constraint) or to different clusters (cannot-link constraint). Even though lacking a full understanding of the clustering structure, this allows the user to partly specify her intuition by indicating for selected object pairs their pairwise cluster relation. Since in many cases these user constraints express a belief rather than certainty, we use the concept of soft constraints, where mistakes (e.g., disagreeing constraints) are possible and a complete compliance of the clustering result with all constraints is not enforced.

The transfer of the semi-supervised clustering principle to the multi-view case poses a severe challenge, particularly regarding the multi-faceted nature of the data. One user might, for example, see the similarity of two movies based on their cast, while another user might foreground their dissimilarity based on differing genres. It, therefore, might remain unclear to which view specific constraints refer to. In particular, when naively assigning all constraints to a single view, a large proportion of the constraints might be conflicting such that even a relaxation to soft constraints will not be sufficient anymore. Therefore, the challenge with semi-supervised multi-view clustering is not only to optimize the clustering such that constraints are optimally fulfilled but also to *learn the affiliation of constraints to views*.

It has to be highlighted that some of the *sequentially* working multi-view clustering approaches (which iteratively find one clustering at a time) (e.g., [BB06,

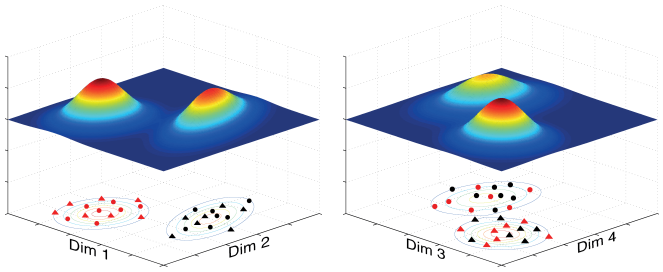


Figure 8.1: Example for the multi-view scenario

QD09]) already work based on instance level constraints to incorporate the feedback of rejected prior clusterings via cannot-link constraints. These constraints, however, are used for a different goal: they guide the clustering method to find *a single* new clustering. Thus, all constraints need to refer to this single clustering, and none of the previous clusterings can be affected by these constraints. In contrast, our aim is to incorporate instance level constraints which might improve the *overall* result of all clusterings. It becomes apparent, that in this case, we have to rely on a clustering technique which detects all clusterings/views *simultaneously*.

Only few approaches for simultaneous multi-view clustering have been proposed (e.g., [NDJ10, JMD08, GMFS09, GFS12]). Here, the inevitable connection of multi-view clustering and subspace clustering has been observed first [NDJ10, GMFS09, GFS12], which later also influenced sequentially working approaches like [DB13b]. Subspace clustering assumes each cluster to have an individual set of relevant data attributes, which corresponds well with the motivation of multi-view clustering that different views on the data (i.e., considering different characteristics of the data) might reveal different clustering structures.

In this work, we join the three paradigms of simultaneous multi-view clustering, subspace clustering, and constraint-based clustering. We present a Bayesian framework that models the different clustering views via several multivariate mixture distributions located in subspace projections (cf. Figure 8.1). Each object follows multiple components, each in a different mixture model, each defining a distribution only for a certain view (i.e., subspace) of the data, and each representing a different role of the object. We integrate the optimal fulfillment of user provided instance level constraints into the Bayesian learning process, where we tackle the challenge of automatically learning the responsibility of views for specific constraints. Our contributions are:

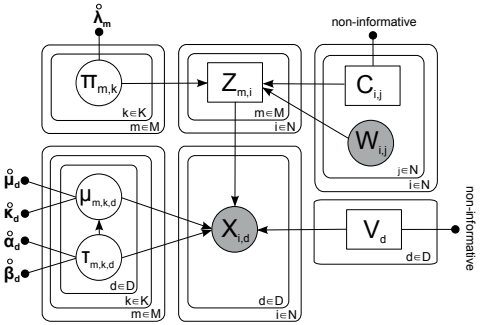


Figure 8.2: Graphical model of our method. Rectangles denote discrete random variables, circles continuous random variables, and black dots (deterministic) hyperparameters of the prior distributions.

- *Multiple clusterings:* We propose a sound Bayesian model which represents multiple clusterings via individual mixture models, each representing a distinct view.
- *Semi-supervision:* Our model incorporates prior knowledge in form of (soft) must-link and cannot-link instance level constraints. Our method automatically learns the assignment of these constraints to specific views if their responsibility is not explicitly specified.
- *Algorithm design:* We present an efficient algorithm based on the principle of variational inference for learning our model.
- *Effectiveness:* We analyze the effectiveness of our method and show its potential to increase the clustering quality by using prior knowledge.

8.2 Bayesian Framework

In this section, we introduce a Bayesian framework for semi-supervised multi-view clustering. An overview of our framework is given by the graphical model depicted in Fig. 8.2. While this section introduces the generative process of our model, we describe in Section 8.3 how to learn the model's parameters given a set of observations. Following convention, we do not distinguish between a random variable x and its realization $x = \mathbf{x}$ if it is clear from the context. As an abbreviation, we denote sets of random variables with the index $*$, e.g., $y_{*,d}$ is the set of random variables $\{y_{i,d}\}$ with i in the corresponding index domain, and Y is an abbreviation for the set $y_{*,*}$.

The number of objects is denoted with N, the number of dimensions with D, the number of clusters/components with K, and the number of alternative views/clusterings with M. We write $k \in K$, as a shortcut for $k \in \{1, \ldots, K\}$.

Multiple Mixture Models The general idea of our method is to represent the multiple clusterings of the data by multiple mixture models, each located in a different subspace projection (cf. Fig. 8.1). In this work, we focus on Gaussian mixture models; extensions to other distributions are straightforward. Following standard principles, each of the M mixture models is based on K components, where each of these components is associated with a mean and a covariance/-precision matrix. To reduce the number of parameters to be estimated, we focus on diagonal precision matrices. Thus, for a Bayesian treatment, we introduce the random variables

$$(\mu_{m,k,d}, \tau_{m,k,d}) \sim \mathcal{NG}(\mathring{\mu}_d, \mathring{\kappa}_d, \mathring{\alpha}_d, \mathring{\beta}_d) \tag{8.1}$$

where $\mu_{m,k,d}$ is the mean of component k in dimensions d for clustering m, and $\tau_{m,k,d}$ the corresponding precision. We select the normal-gamma distribution $\mathcal{NG}$ as a prior since it represents the variables' conjugate prior. The hyperparameters denoted by $\mathring{*}$ can be used to control the mixture models' components if some prior knowledge is available. Per default, we choose least informative priors by selecting $\mathring{\kappa}_d, \mathring{\alpha}_d \to 0$ and setting $\mathring{\mu}_d / \mathring{\beta}_d$ to be the sample mean/sum of squared deviations in dimension d.

Besides the components parameters, each mixture model is associated with a corresponding random variable representing the mixture weights. Obviously, since we want to find multiple different clusterings, these weights can be different for each view. We use the random variable

$$\vec{\pi}_m \sim Dir(\mathring{\lambda}) \tag{8.2}$$

where $\pi_{m,k}$ is the weight of component k in clustering m. Due to conjugate properties, we use a Dirichlet distribution as its prior. Again, in our study, we use a non-informative prior by selecting $\mathring{\lambda} = 1$ since a priori no knowledge about the cluster sizes is given.

Integrating Subspaces To detect the data's multiple views, we refer to the principle of subspace clustering. Our goal is to assign each mixture model to a specific subspace projection, which it describes well. Since the relevant dimensions of the mixtures are a priori not known, we learn them with our method. Therefore, we introduce the random variable

$$v_d \sim Categorical(\vec{r}_d) \tag{8.3}$$

to indicate which of the M clusterings is responsible for a specific dimension d. The vector $\vec{r}_d \in [0 \dots 1]^M$ (with $\sum r_{k,m} = 1$) can be used to give some prior knowledge which dimension belongs to which view. Again, we use a constant non-informative prior, i.e., $r_{k,m} = 1/M$.

Knowing about the subspaces as well as the mixture models' parameters, we are now able to generate observations which show multiple clustering structures: We denote with $z_{m,i}$ the random variable indicating to which cluster an object i belongs to in clustering m, i.e.,

$$z_{m,i} \sim Categorical(\vec{\pi}_m) \tag{8.4}$$

Note that for each view m, the object might follow a different cluster, i.e., $z_{m,i} \neq z_{m',i}$ is possible. Thus, in each view the object might be grouped together with different objects. This idea is illustrated in Figure 8.1: the grouping on the left differs from the one on the right. Given $z_{m,i}$, the attribute value of object i in dimension d is drawn according to

$$x_{i,d} \sim \mathcal{N}(\mu_{m,k,d}, \tau^{-1}_{m,k,d}) \text{ with } m = v_d \text{ and } k = z_{m,i} \tag{8.5}$$

That is, we use the clustering m which is responsible for dimensions d and the corresponding component k the object belongs to in this view.

Integrating User Constraints So far, our model corresponds to a completely unsupervised technique for finding multiple clusterings. As a major advancement, we now integrate user provided prior-knowledge. As discussed, we aim at supporting the concept of instance level constraints. More precisely, we support the idea of soft constraints between pairs of objects that indicate whether the objects should or should not be grouped together. We selected this type of semi-supervision since it reflects an intuitive understanding of clustering and is easy to specify for the user.

The user can provide a constraint between the objects i and j via a weight $w_{i,j}$. If the weight is positive, the user indicates that there should exist a clustering where the objects are grouped together. If the weight is negative, the user indicates that there should exist a clustering where i and j are not grouped together. Different magnitudes of the weights can be used to indicate the different importance or relevance of the constraints.

At this point, it is crucial to keep in mind that we are interested in finding multiple, alternative clusterings: A constraint between i and j means that *there*

exists a view where the constraint is fulfilled. We do not require that i and j are grouped together in *all views*, which actually would contradict the fundamental assumption for multi-view scenarios that clusterings of different views differ and contain alternative knowledge. Forcing constraints to be valid for all views would be too restrictive. Furthermore, we argue that the user is generally not aware of the details of all possible groupings. Thus, the user constraints should not be able to restrict views that the user does not yet understand. Accordingly, for each constraint, we are interested in finding (at least) one clustering fulfilling this constraint.

Resulting from this principle, another challenge of our method becomes apparent: we have to determine the clustering which is responsible for a specific constraint. In the following, we show how to model all these aspects.

As mentioned, the constraints are modeled via weights. In our model, we represent them via a symmetric matrix W of size $N \times N$, where entries with weight zero indicate no prior knowledge about the corresponding pairs of objects. In practice, we can use a *sparse representation* of the matrix which only encodes the given constraints and allows for an efficient processing. Interesting to note is that the (observed) matrix W appears in our grapical model as one of the root nodes (cf. Fig. 8.2), and not as a leaf like X. As shown, the weights influence the grouping Z of the objects.

Additionally, we introduce the categorical random variables $c_{i,j}$ (due to the symmetry of the weights, we only need to consider $i < j$). These variables indicate which view is responsible for a specific constraint. That is, we have

$$c_{i,j} \sim Categorical(\vec{h}^{(i,j)}) \tag{8.6}$$

where $\vec{h}^{(i,j)} \in [0...1]^M$ with $\sum_{m \in M} h_m^{(i,j)} = 1$. The user can use $\vec{h}^{(i,j)}$ to express some further prior knowledge about the constraint between object i and j. If the user, for example, knows that a set of constraints should most likely belong to one view, the h vectors can be selected accordingly. Per default, we assume that no knowledge about the assignment of constraints to views is known, i.e., we use $h_m^{(i,j)} = 1/M$.

Given W and C, how can we use their values to influence the clustering structure of the data? Our idea is to add a bias to the probability distribution of the $z_{m,j}$. The probability of generating a clustering that matches the constraints should be higher than the probability of a clustering which violates the constraints. Particularly, this results in a dependency between the variables $z_{m,*}$

which is guided by the constraints. We define

$$p(z_{m,*} \mid \vec{\pi}_m, W, C) \propto \prod_{i=1}^{N} \pi_{m,z_{m,i}} \cdot \prod_{i=1}^{N} \prod_{\substack{j>i \\ c_{i,j}=m}}^{N} e^{w_{i,j} \cdot \delta(z_{m,i}, z_{m,j})} \tag{8.7}$$

Here, $\delta(z_{m,i}, z_{m,j})$ denotes the Kronecker delta, which evaluates to 1 if both objects are located in the same cluster (in view m), and 0 otherwise. Please note that Equation 8.7 is the joint distribution for all $z_{m,*}$.

The first part of the equation corresponds to the mixture weights as used in standard mixture models. If all $w_{i,j} = 0$, Equation 8.4 and 8.7 are equivalent. The second part models the bias to specific groupings: As one can see, if $w_{i,j}$ is positive and the objects are located in the same cluster, the probability of selecting this grouping increases. Accordingly, if $w_{i,j}$ is negative, one would decrease the probability of clusterings where i and j are grouped together. A similar principle was used in [LL04, BBM04b] for single-view clustering.

Important to mention is that the second part of the equation incorporates the automatic assignment of constraints to views. The constraint between i and j adds a bias to the clustering structure in view $c_{i,j} = m$ only. In accordance to our discussion above, the other views are not affected.

Given the new definition for the distribution of Z, the actual observations are, as before, generated according to Equation 8.5. Overall, our model combines the principle of multiple clusterings in subspace projections with the paradigm of semi-supervised clustering and automatically assigns constraints to their responsible views.

8.3 The SMVC Algorithm

While the previous section has focused on the model's generative process, we now present our learning technique. That is, *given* a set of observations X and a set of constraints W, we infer the model's parameters. Our method is called SMVC (Semi-Supervised Multi-View Clustering).

8.3.1 Variational Inference

The general inference problem we have to solve is to determine the distribution $p(Y|X, W)$, where $Y = \{V, Z, C, \vec{\pi}, \mu, \tau\}$ is the set of all latent variables. Based

on this distribution, we can, e.g., pick the realizations of the latent variables leading to the highest likelihood given the data. Since computing $p(Y|X,W)$ is intractable, we compute an approximation based on the principle of variational inference [Bis06]: we approximate $p(Y|X,W)$ by a tractable family of parametrized distributions $q(Y|\Psi)$. The parameters Ψ are the free variational parameters. These parameters are optimized such that the best approximation between q and p is obtained. Technically, one minimizes the Kullback-Leibler divergence between q and p by optimizing Ψ. Using Jensen's inequality, minimizing the KL divergence is equivalent to maximizing the following lower bound on the log marginal likelihood [Bis06]:

$$\mathcal{L}(X,W;\Psi) = \mathbb{E}_q[\ln p(X,W,Y)] - \mathbb{E}_q[\ln q(Y|\Psi)] \tag{8.8}$$

where $\mathbb{E}_q[.]$ denotes the expectation w.r.t. the q distribution.

Following primarily the idea of mean field approximation, we assume the function q to factorize in

$$\begin{aligned} p(Y \mid X,W) \approx q(Y|\Psi) := & \prod_d q_1(v_d) \cdot \prod_m \prod_i q_2(z_{m,i}) \\ & \cdot \prod_i \prod_{j>i} q_3(c_{i,j}) \cdot \prod_m q_4(\vec{\pi}_m) \cdot \prod_m \prod_k \prod_d q_5(\mu_{m,k,d}, \tau_{m,k,d}) \end{aligned}$$

As we will later see, assuming the above factorization, the optimal variational distributions have the form

$$\begin{aligned} q_1(v_d) &= Categorical(v_d \mid \phi_{d,1}, ..., \phi_{d,M}) \\ q_2(z_{m,i}) &= Categorical(z_{m,i} \mid \psi_{m,i,1}, ..., \psi_{m,i,K}) \\ q_3(c_{i,j}) &= Categorical(c_{i,j} \mid \xi_{i,j,1}, ..., \xi_{i,j,M}) \\ q_4(\vec{\pi}_m) &= Dir(z_{m,i} \mid \vec{\lambda}_m) \\ q_5(\mu_{m,k,d}, \tau_{m,k,d}) &= \mathcal{NG}(\mu_{m,k,d}, \tau_{m,k,d} \mid \\ & \qquad \tilde{\mu}_{m,k,d}, \tilde{\kappa}_{m,k,d}, \tilde{\alpha}_{m,k,d}, \tilde{\beta}_{m,k,d}) \end{aligned}$$

where $\Psi = \{\phi, \psi, \xi, \vec{\lambda}, \tilde{\mu}, \tilde{\kappa}, \tilde{\alpha}, \tilde{\beta}\}$ are the variational parameters to be optimized. Note that each distribution has its own variational parameters [Bis06]. Thus, e.g., the functions $q_1(v_d)$ and $q_1(v_{d'})$, are not necessarily identical. This extra degree of freedom allows to find a good approximation between q and p. As discussed in Section 8.2, for $c_{i,j}$, i.e., the function q_3, we only need to consider pairs i,j with $w_{i,j} \neq 0$.

For the variational distributions, the following holds:

$$\mathbb{E}_q[\![z_{m,i} = k]\!] = \psi_{m,i,k} \qquad \mathbb{E}_q[\![c_{i,j} = m]\!] = \xi_{i,j,m} \qquad \mathbb{E}_q[\![v_d = m]\!] = \phi_{d,m}$$

$$\mathbb{E}_q[\pi_{m,k}] = \frac{\tilde{\lambda}_m[k]}{\sum_{i=1}^{K} \tilde{\lambda}_m[i]} \qquad \mathbb{E}_q[\log \pi_{m,k}] = \psi(\tilde{\lambda}_m[k]) - \psi(\sum_{i=1}^{K} \tilde{\lambda}_m[i])$$

$$\mathbb{E}_q[\mu_{m,k,d}] = \tilde{\mu}_{m,k,d} \qquad \mathbb{E}_q[\mu_{m,k,d} \cdot \tau_{m,k,d}] = \tilde{\mu}_{m,k,d} \cdot \frac{\tilde{\alpha}}{\tilde{\beta}}$$

$$\mathbb{E}_q[\tau_{m,k,d}] = \frac{\tilde{\alpha}}{\tilde{\beta}} \qquad \mathbb{E}_q[\log \tau_{m,k,d}] = \psi(\tilde{\alpha}) - \log(\tilde{\beta})$$

$$\mathbb{E}_q[\mu^2_{m,k,d} \cdot \tau_{m,k,d}] = \frac{1}{\tilde{\kappa}_{m,k,d}} + \tilde{\mu}^2_{m,k,d} \cdot \frac{\tilde{\alpha}}{\tilde{\beta}}$$

General Processing Scheme We use an iterative coordinate ascent method to maximize Equation 8.8 w.r.t. the parameters Ψ (the update equations follow in Section 8.3.2). The processing scheme is as follows:

```
1 while not converged do
2     for i, j ∈ N : j > i ∧ w_{i,j} ≠ 0 do update ξ_{i,j,*}      Eq. 8.10
3     for d ∈ D do update φ_{d,*}                                  Eq. 8.11
4     for m ∈ M, i ∈ N do update ψ_{m,i,*}                         Eq. 8.12
5     for i ∈ N, m ∈ M do update λ⃗_m                               Eq. 8.13
6     for m ∈ M, k ∈ K, d ∈ D do                                   Eq. 8.14
7         update μ̃_{m,k,d}, κ̃_{m,k,d}, α̃_{m,k,d}, β̃_{m,k,d}
```

Note that due to the properties of variational inference [Bis06], it is guaranteed that the method converges. In practice, we assume convergence if the change in the lower bound on the marginal likelihood is below 0.01. Additionally, to avoid the problem of local minima, we enhance the processing scheme by gradually increasing the importance of the constraints. That is, starting with low weights, we linearly increase the values $w_{i,j}$ until they reach the user specified scores. This way, the constraints do not force the undeveloped clustering in misleading directions but the constraints' influence increases gradually to guide the clustering as it evolves. For initializing our method, we exploit the same principle as described in [GFS12]. The random variable C/q_3 is initialized randomly based on its prior distribution.

8.3.2 Update Equations

We briefly present the update equations required for the coordinate ascent method. We primarily follow the principle of [Bis06]: The optimal distribution for $q_x(B)$ can be determined by

$$\ln q_x^*(B) = \mathbb{E}_{q\setminus B}[\ln p(X, Y, W)] + \mathcal{C} \tag{8.9}$$

Here, the constant $\mathcal{C}$ absorbs all terms which are independent of B and, thus, do not affected the optimal distribution of q_x. $\mathbb{E}_{q\setminus B}[.]$ denotes the expectation w.r.t. the distribution q taken over all variables Y except of B. To avoid cluttering the notation, we simply write $\mathbb{E}_q$ in the following (it is clear from the context which variable is excluded).

Updating the constraint responsibility Let $[\![.]\!]$ denote the Iverson bracket. We can rewrite Equation 8.7 as follows

$$\prod_{i=1}^{N}\prod_{k=1}^{K} \pi_{m,k}^{[\![z_{m,i}=k]\!]} \cdot \prod_{i=1}^{N}\prod_{j>i}^{N}\prod_{k=1}^{K} e^{w_{i,j}[\![z_{m,i}=k]\!][\![z_{m,j}=k]\!][\![c_{i,j}=m]\!]}$$

This formulation makes it easier to derive the following results. Accordingly, we can rewrite the remaining equations.

The optimal distribution for $q_3(c_{a,b})$ (with $a < b$) can be obtained via Equation 8.9. Removing all terms which are independent of $c_{a,b}$ and using the above reformulation, we get

$$\begin{aligned}
&\log q_3^*(c_{a,b} = y) \\
&= \mathbb{E}_q[\log\left(P(c_{i,j})P(Z|\pi, C, W)\right)] + \mathcal{C} \\
&= \mathbb{E}_q[\log\frac{1}{M}] + \mathbb{E}_q[\log\prod_{m=1}^{M}\Bigg(\prod_{i=1}^{N}\prod_{k=1}^{K}\pi_{m,k}^{[\![z_{m,i}=k]\!]} \cdot \prod_{i=1}^{N}\prod_{j>i}^{N} \\
&\qquad\qquad \cdot \prod_{k=1}^{K} e^{w_{i,j}[\![z_{m,i}=k]\!][\![z_{m,j}=k]\!][\![c_{i,j}=m]\!]}\Bigg)] + \mathcal{C} \\
&= \mathbb{E}_q[\sum_{m=1}^{M}\sum_{k=1}^{K}\log e^{w_{a,b}[\![z_{m,a}=k]\!][\![z_{m,b}=k]\!][\![c_{a,b}=m]\!]}] + \mathcal{C} \\
&= w_{a,b}\sum_{k=1}^{K}\mathbb{E}_q[\![z_{y,a}=k]\!] \cdot \mathbb{E}_q[\![z_{y,b}=k]\!] + \mathcal{C}
\end{aligned}$$

Since $c_{a,b}$ has a finite domain, the distribution q_3 is a categorical distribution. Renaming the variables, the optimal hyperparameters of the distribution $q_3(c_{i,j})$ are given by

$$\xi_{i,j,m} \propto exp^{\left(w_{i,j} \sum_{k=1}^{K} \mathbb{E}_q[\![z_{m,i}=k]\!] \cdot \mathbb{E}_q[\![z_{m,j}=k]\!]\right)} \tag{8.10}$$

where $\sum_m \xi_{i,j,m} = 1$. The occurring expectations can be replaced by the known expectations of the variational distributions. Intuitively, the parameter $\xi_{i,j,m}$ reflects the probability of assigning the constraint between i and j to the view m.

Updating the views Computing Equation 8.9 for $q_1(v_d)$ and removing all terms which are independent of v_d leads to

$$\begin{aligned}
&\ln q_1^*(v_d = y) \\
&= \mathbb{E}_q[\log\left(P(x_{*,d}|v_d, Z, \mu, \tau)P(v_d)\right)] + \mathcal{C} \\
&= \mathbb{E}_q[\log \prod_{m=1}^{M}\prod_{i=1}^{N}\prod_{k=1}^{K} \mathcal{N}(x_{i,d}|\mu_{m,k,d}, \tau_{m,k,d}^{-1})^{[\![v_d=m]\!][\![z_{m,i}=k]\!]}] + \mathbb{E}_q[\log \frac{1}{M}] + \mathcal{C} \\
&= \mathbb{E}_q[\log \prod_{i=1}^{N}\prod_{k=1}^{K} \mathcal{N}(x_{i,d}|\mu_{y,k,d}, \tau_{y,k,d}^{-1})^{[\![z_{y,i}=k]\!]}] + \mathcal{C} \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{E}_q[\![z_{y,i}=k]\!] \cdot f(y,k,d,i) + \mathcal{C}
\end{aligned}$$

Here, we used the definition

$$\begin{aligned}
&f(m,k,d,i) := \mathbb{E}_q[\mathcal{N}(x_{i,d}|\mu_{m,k,d}, \tau_{m,k,d}^{-1})] \\
=&\mathbb{E}_q[\log \sqrt{\frac{\tau_{y,k,d}}{2\pi}} e^{\frac{-(x_{i,d}-\mu_{y,k,d})^2\tau_{y,k,d}}{2}}] \\
=&\frac{1}{2}\mathbb{E}_q[\log \frac{\tau_{y,k,d}}{2\pi}] + \frac{1}{2}\mathbb{E}_q[-(x_{i,d}-\mu_{y,k,d})^2\tau_{y,k,d}] \\
=&\frac{1}{2}\cdot(\mathbb{E}_q[\log \tau_{y,k,d}] - x_{i,d}^2 \cdot \mathbb{E}_q[\tau_{y,k,d}] + 2 \cdot x_{i,d} \cdot \mathbb{E}_q[\mu_{y,k,d} \cdot \tau_{y,k,d}] \\
&\qquad - \mathbb{E}_q[\mu_{y,k,d}^2 \cdot \tau_{y,k,d}] - \mathbb{E}_q[\log 2\pi])
\end{aligned}$$

Thus, q_1 is a categorical distribution and the optimal hyperparameters for $q_1(v_d)$ are given by

$$\phi_{d,m} \propto exp^{\sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{E}_q[\![z_{m,i}=k]\!] \cdot f(m,k,d,i)} \tag{8.11}$$

where $\sum_m \phi_{d,m} = 1$.

Updating the cluster indicator The same principle can be applied for the cluster indicator variable. We obtain:

$$
\begin{aligned}
&\log q_2^*(z_{m,a} = y) \\
&= \mathbb{E}_q[\log\left(P(x_{a,*}|V, Z, \mu, \tau)P(Z|\pi, C, W)\right)] + \mathcal{C} \\
&= \mathbb{E}_q[\log \prod_{d=1}^{D}\prod_{k=1}^{K} \mathcal{N}(x_{a,d}|\mu_{m,k,d}, \tau_{m,k,d}^{-1})^{[\![v_d=m]\!][\![z_{m,a}=k]\!]}] + \\
&\qquad\qquad \mathbb{E}_q[\log \prod_{k=1}^{K}\prod_{i=1}^{N} \pi_{m,k}^{[\![z_{m,i}=k]\!]} \prod_{i=1}^{N}\prod_{j>i}^{N} e^{w_{i,j}[\![z_{m,i}=k]\!][\![z_{m,j}=k]\!][\![c_{i,j}=m]\!]}] \\
&= \sum_{d=1}^{D} \mathbb{E}_q[\![v_d = m]\!]\, \mathbb{E}_q[\log \mathcal{N}(x_{a,d}|\mu_{m,y,d}, \tau_{m,y,d}^{-1})] + \mathbb{E}_q[\log \pi_{m,y}] + \\
&\qquad\qquad \sum_{i=1}^{N}\sum_{j>i}^{N} w_{i,j}\, \mathbb{E}_q[\![z_{m,i} = y]\!]\, \mathbb{E}_q[\![z_{m,j} = y]\!]\, \mathbb{E}_q[\![c_{i,j} = m]\!] + \mathcal{C} \\
&= \sum_{d=1}^{D} \mathbb{E}_q[\![v_d = m]\!] \cdot f(m, y, d, a) + \\
&\qquad\qquad \mathbb{E}_q[\log \pi_{m,y}] + \sum_{j\neq a}^{N} w_{a,j}\, \mathbb{E}_q[\![z_{m,j} = y]\!]\, \mathbb{E}_q[\![c_{a,j} = m]\!] + \mathcal{C}
\end{aligned}
$$

Here, we exploit the symmetry of $w_{i,j}$ and the definition of f as given above. Note again, that we do not actually need to sum over all $j \neq a$ when using a sparse encoding of the matrix W. It is sufficient to iterate over those j for which a constraint with a is given. Similar as before, the optimal hyperparameters for $q_2(z_{m,i})$ are given by

$$
\begin{aligned}
\psi_{m,i,k} \propto exp\Big(&\sum_{d=1}^{D} \mathbb{E}_q[\![v_d = m]\!] \cdot f(m, k, d, i) \\
&+ \mathbb{E}_q[\log \pi_{m,k}] + \sum_{j\neq i}^{N} w_{i,j}\, \mathbb{E}_q[\![z_{m,j} = k]\!]\, \mathbb{E}_q[\![c_{i,j} = m]\!]\Big) \quad (8.12)
\end{aligned}
$$

with $\sum_k \psi_{m,i,k} = 1$.

Updating the mixing weights The mixing weights are continuous. Since we selected a conjugate prior in our model, it follows:

$$
\begin{aligned}
&\log q_4^*(\vec{\pi}_m) \\
&= \mathbb{E}_q[\log\left(P(\pi_m)P(z_{m,*}|\pi,C,W)\right)] + \mathcal{C} \\
&= \mathbb{E}_q[\log\left(\frac{\Gamma(\mathring{\lambda}K)}{\Gamma(\mathring{\lambda})^K}\prod_{k=1}^{K}\pi_{m,k}^{\mathring{\lambda}-1}\right)] + \mathbb{E}_q[\log\left(\prod_{i=1}^{N}\prod_{k=1}^{K}\pi_{m,k}^{[\![z_{m,i}=k]\!]}\prod_{i=1}^{N}\right. \\
&\qquad\qquad \left.\cdot\prod_{j>i}^{N}\prod_{k=1}^{K}e^{w_{i,j}[\![z_{m,i}=k]\!][\![z_{m,j}=k]\!][\![c_{i,j}=m]\!]}\right)] + \mathcal{C} \\
&= \sum_{k=1}^{K}(\mathring{\lambda}-1)\mathbb{E}_q[\log\pi_{m,k}] + \sum_{i=1}^{N}\sum_{k=1}^{K}\mathbb{E}_q[\![z_{m,i}=k]\!]\mathbb{E}_q[\log\pi_{m,k}] + \mathcal{C} \\
&= \sum_{k=1}^{K}\left((\mathring{\lambda}-1) + \sum_{i=1}^{N}\mathbb{E}_q[\![z_{m,i}=k]\!]\right)\cdot\mathbb{E}_q[\log\pi_{m,k}] + \mathcal{C}
\end{aligned}
$$

As seen, the optimal distribution for q_4 is a Dirichlet distribution, where the hyperparameters are given by

$$\tilde{\lambda}_m[k] = \mathring{\lambda} + \sum_{i=1}^{N}\mathbb{E}_q[\![z_{m,i}=k]\!] \tag{8.13}$$

Updating the mixture components Updating the mean and precision of each mixture component follows the standard principle of variational inference in a conjugate setting.
Let $u_{m,k} = \sum_{i=1}^{N}\mathbb{E}_q[\![z_{m,i}=k]\!]$ be the unnormalized weight of a cluster and $\bar{x}_{m,k,d} = \frac{1}{u_{m,k}}\sum_{i=1}^{N}x_{i,d}\mathbb{E}_q[\![z_{m,i}=k]\!]$ its weighted mean in dimension d (when considering the expectation w.r.t. q). Using conjugacy, it follows that the optimal hyperparameters of the distribution q_5 are given by

$$
\begin{gathered}
\tilde{\mu}_{m,k,d} = \frac{\mathring{\kappa}_d\,\mathring{\mu}_d + u_{m,k}\,\bar{x}_{m,k,d}}{\mathring{\kappa}_d + u_{m,k}} \qquad \tilde{\kappa}_{m,k,d} = \mathring{\kappa}_d + u_{m,k} \qquad \tilde{\alpha}_{m,k,d} = \mathring{\alpha}_d + \frac{u_{m,k}}{2} \\
\tilde{\beta}_{m,k,d} = \mathring{\beta}_d + \frac{1}{2}\sum_{i=1}^{N}(x_{i,d}-\bar{x}_{m,k,d})^2 + \frac{\mathring{\kappa}_d\,u_{m,k}}{\mathring{\kappa}_d + u_{m,k}}\frac{(\bar{x}_{m,k,d}-\mathring{\mu}_d)^2}{2}
\end{gathered} \tag{8.14}
$$

8.3.3 Complexity and Summary

Inspecting the individual update equations, it becomes apparent that each iteration of our algorithm runs in time $\mathcal{O}(M \cdot N \cdot K \cdot (D + W))$, where W denotes the number of constraints. Thus, we obtain a *linear complexity* in all important parameters.

Overall, our method efficiently computes an approximation of the posterior distribution $p(Y|X, W)$ which shows us the multiple clustering structures, their relevant subspaces, and the assignment of constraints to views.

8.4 Related Work

Our approach is related to four main paradigms in the field of cluster analysis: subspace clustering, multi-view clustering, model-based clustering, and semi-supervised clustering. Since the first three paradigms have already been discussed in Chapter 2 and in the related work section of Chapter 7, we will mainly focus on the most related approaches presented in the field of semi-supervised clustering. Table 8.1 shows an overview of the related work and their corresponding properties.

In Chapter 2, we differentiated between two paradigms for multi-view clustering: iterative and simultaneous approaches. Approaches that iteratively determine a new clustering based on previous results can partially be categorized as semi-supervised, since previous clustering solutions serve as guidance for the discovery of new clustering structures. However, the constraints affect only the solution of the single, next clustering and, thus, already detected solutions cannot benefit from them. Furthermore, all approaches presented for the iterative paradigm utilizing space transformations [CFD07, DQ08, QD09, DB13b] suffer from distortions of the original space, which hinder an intuitive interpretation of the clustering result.

Semi-supervised clustering The detection and usefulness of multiple clustering solutions strongly depends on the user's preferences. For different users and applications, different clusterings might prove to be useful. Semi-supervised clustering [BDW08] provides a possibility to accommodate these preferences as additional information or domain knowledge into the clustering process. For traditional single view, full-space clustering (e.g., k-Means) a popular solution is to use instance level constraints: the objective function is extended by penalizing

	Multi-View	Simultaneous processing	Subspace projections	Semi-supervised	Constraint responsibility
Subspace clustering	–	✓	✓	–	–
Multi-view clustering					
↪ iterative	✓	–	–	○	fixed
↪ simultaneous	✓	✓	✓	–	–
Semi-supervised clust.	–	✓	–	✓	fixed
Our method	✓	✓	✓	✓	learned

Table 8.1: Overview of related paradigms

violated constraints [BBM04a] or one learns a distance metric that best represents the constraints [BBM04c]. For model-based clustering, few extensions for equivalence constraints exist. [SBHHW03] introduces a closed form EM based on the transitive closure of must-link constraints and proposes a Markov network for handling cannot-link constraints. Since it neither can incorporate both constraint types simultaneously nor cope with conflicting constraints, [LL04, BBM04b] propose to integrate negative and positive pairwise constraints as priors into Gaussian mixture models, which allows for modeling soft as well as hard constraints. These approaches have shown to substantially improve the clustering result in the single view case. Since in the multi-view case, we are uncertain which constraints refer to which view, these existing solutions cannot easily be transferred.

Methods such as [Agg04] use supervision (e.g., human interaction) to enhance the clustering in a single given subspace. In contrast, we exploit supervision to enhance the clustering result across all views simultaneously. Works such as [GBS12a] combine subspace clustering with graph clustering. The underlying graph might be regarded as a certain type of supervision. These methods do not focus on finding alternative groupings in the attribute space.

Overall, none of the existing approaches is able to incorporate prior information for a multi-view clustering solution, where constraints may refer to different clustering views. Our new statistical model handles different clustering views in different attribute subspaces and learns responsibilities of views for the provided equivalence constraints.

8.5 Experimental Analysis

Setup We compare SMVC with representatives from all three paradigms: multi-view clustering, subspace clustering, and semi-supervised clustering. For multi-view clustering, we choose the four approaches *Multi-View 1* and *Multi-View 2* proposed in [CFD07], the *Alternative Clustering* method proposed in [QD09], and our *MVGen* [GFS12] approach. These approaches best reflect the demands for multi-view clustering as discussed in Section 8.4. As subspace clustering approaches, we choose the partitioning approach *Proclus* and *StatPC*, which allows for overlapping clusters. Furthermore, we compare against the two semi-supervised approaches *PCKMeans* [BBM04a] and *MPCKMeans* [BBM04c], both using instance level constraints.

For case studies on real world data, we use the CMUFaces, Iris, and Wine data (all from the UCI repository [FA10]), and drawn stick figures. Synthetic data containing multiple views is generated based on our generative model. The default dataset contains 2 disjoint views, each with 4 clusters, 20 dimensions, and 5000 objects.

Each method is provided with the number m_{max} of views and the number k_{max} of clusters per view. If the algorithm does not allow for setting these parameters, we choose the default parameter setting.

Runtime is measured on 4GHz AMD FX-8350 CPU with 16 GB main memory. Quality is assessed based on the $E4SC$ measure (cf. Chapter 13), which is a symmetric and subspace aware variant of the popular F1 measure. Since most of the competing approaches do not determine axis parallel subspaces, we refrain from evaluating the subspaces and just concentrate on the object groupings (for clarity, we rename the measure to 'E4FC'). For all quality experiments, we average the results over ten executions.

8.5.1 Evaluation on Synthetic Data

Varying number of constraints We start our evaluation by examining the influence of a varying number of constraints in Figure 8.3. Here, we tested three different variants of the semi-supervised clustering approaches: We either used only must-link constraints (SMVC-ML), only cannot-link constraints (SMVC-CL), or a combination of 50% from both (SMVC-Comb). Note that in this experiment, we randomly generated constraints based on the ground truth clusters known for synthetic data. These constraints might not help to improve the clustering and,

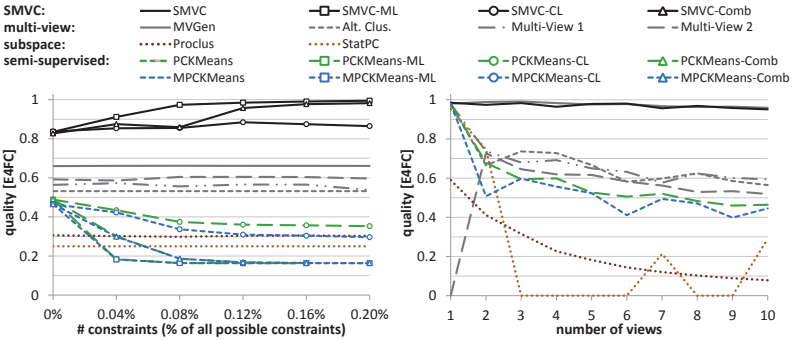


Figure 8.3: Quality vs. # constraints

Figure 8.4: Quality vs. # views

thus, represent only very weak supervision. In practice, the user might provide better constraints, e.g., via the principles of active learning [BBM04a], where the object pairs to be constrained are actively selected based on their impact for the clustering quality. This way, usually fewer constraints are required to achieve a better clustering result than with random constraints.

Figure 8.3 shows the results for an increasing number of constraints: Here, we generated a challenging dataset with clusters having a large variance to study the benefit of semi-supervision. Most approaches fail to identify a meaningful clustering structure for this difficult clustering scenario. SMVC is not only the approach showing the best clustering results without the help of prior knowledge, it is also the only approach able to improve its clustering based on additional constraints. For the two other semi-supervised approaches PCKMeans and MPCKMeans, we even observe a decreasing clustering quality with increasing amount of prior knowledge! This indicates, that they cannot deal with the potentially disagreeing constraints of the two views.

We furthermore can see the varying influence of the different constraints (100% must-link constraints, 100% cannot-link constraints, or 50% must-link + 50% cannot-link). The higher the proportion of must-link constraints, the higher is the influence. The reason is that cannot-link constraints a priori have a higher possibility to be fulfilled than must-link constraints (for m views, each with k clusters, the probability to fulfill a cannot-link constraint is $m \cdot \binom{k}{2}$, whereas for must-link constraints it is $m \cdot k$). Therefore, we will focus on must-link constraints in the following experiments.

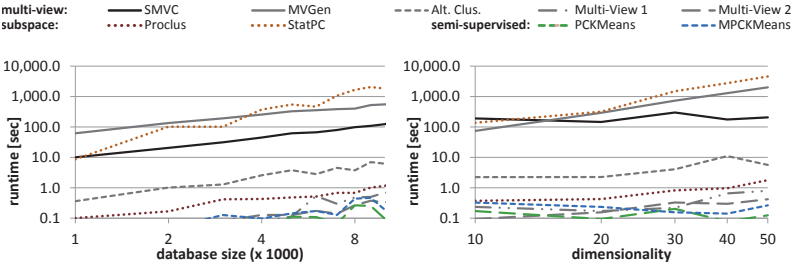


Figure 8.5: Runtime vs. database size

Figure 8.6: Runtime vs. # dimensions

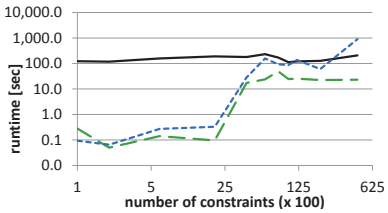


Figure 8.7: Runtime vs. # constraints

Another interesting observation, also stated in [Dav12], is that more constraints do not necessarily result in a better quality. They can even decrease the clustering quality. In Figure 8.3, we can observe this slightly for cannot-link constraints (SMVC-CL); other experiments showed similar effects for must-link constraints. We kindly refer to [Dav12] for a discussion about these effects. Unfortunately, the principles discussed in [Dav12] for wisely choosing the set of constraints are not easily transferable to our scenario.

Varying number of views In the next experiment, we study the potential of using SMVC as an unsupervised technique in a multi-view setting. In Figure 8.4, we vary the number of hidden views in the data. The dimensionality of each view is five, i.e., with increasing number of views, the data's overall dimensionality increases as well. As depicted, SMVC and MVGen are the only approaches able to detect the clustering structure in the case of a large number of views. Their clustering quality is very high and proves to be robust against a varying number of views. The competing methods behave differently: while for single-view data the quality is relatively high, their quality heavily decreases with an increasing number of views.

Scalability Even though the focus for SMVC lies on its clustering quality, we briefly analyze its efficiency. As already discussed in Section 8.3, SMVC scales linearly in the number of objects (Figure 8.5), linearly in the number of dimensions (Figure 8.6), and linearly in the number of constraints (Figure 8.7). Please note the logarithmic scaling of both axes in all three plots. For a varying database size (Figure 8.5), all algorithms show an increasing runtime. The approaches that represent adaptations of the simple and efficient KMeans algorithm (which also includes Proclus) clearly show the lowest runtimes. The runtime of SMVC is comparable to the other algorithms analyzing subspace projections (MVGen, StatPC) and even manages to outperform them thanks to the efficient variational inference techniques.

The benefit of SMVC becomes apparent for a high data dimensionality (Figure 8.6). Due to the exponential number of subspaces, most subspace clustering algorithms (e.g., StatPC) suffer from a tremendously increasing runtime for an increasing number of dimensions. Also MVGen cannot compete with our SMVC due to the complex model selection process. Contrarily, for SMVC, we observe a moderate increase in runtime. This enables us to apply SMVC also on high-dimensional data, as we will see in the experiments on real world data.

Figure 8.7 shows the runtimes of the semi-supervised methods for a varying number of constraints. Here, it is hard to verify the linear runtime of SMVC because constraints support the clustering procedure and, thus, help decreasing the number of iterations. For a small number of constraints, the two KMeans-based approaches can maintain a low runtime. For an increasing number of constraints, however, their runtime eventually even meets the one of SMVC. Of course, such a high number of constraints might not be realistic for most applications.

8.5.2 Evaluation on Real World Data

For evaluation on real world data, we use different evaluation principles, all focusing on the multi-view aspect.

Case study A In Figures 8.8 and 8.9, we extend the datasets Iris and Wine to data containing multiple views: for this, we randomly concatenate the attribute values of different objects up to five times to a higher dimensional space. The original datasets have dimensionalities of 4 and 13, respectively, while the extension to multi-view data leads to dimensionalities up to $5 \cdot 4 = 20$ (Iris) and $5 \cdot 13 = 65$ (Wine).

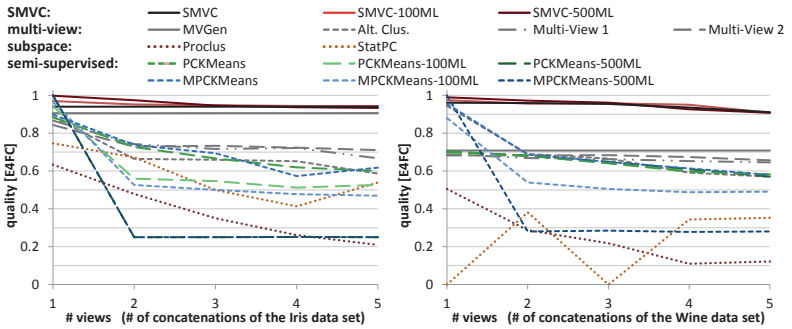


Figure 8.8: Quality on iris data

Figure 8.9: Quality on wine data

For just one view, the quality of some competing approaches is similar to the one of SMVC. However, for an increasing number of views the clustering quality for almost all competing approaches decreases. Only MVGen and SMVC are nearly not affected by an increasing number of views but detect the different object groupings even for multiple views.

To study the effects of semi-supervision, we additionally provided for both datasets 100 and 500 constraints. For just a single view, SMVC is able to improve the cluster quality. On iris, for example, the quality increases from 0.94 over 0.97 to 1.0. The full potential of our approach, however, can bee seen in the case of multiple views: While it is still able to benefit from prior knowledge, the clustering quality of the competing approaches dramatically decreases.

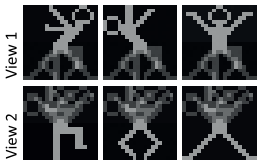
It is noticeable, that with increasing number of views, the constraints seem to have less positive effect on the result of SMVC. This phenomenon can, however, easily be explained by the fact that the constraints have to be distributed among the views, i.e., the proportion of prior knowledge decreases with increasing number of views.

Summarizing, the results for real world data are consistent with the observations made for the synthetic data.

Case study B For our next study, we created a dataset consisting of 900 20x20 images of 'dancing stick figures'. This dataset allows an easy visual interpretation of the clustering results. We drew 9 basic stick figures (Figure 8.10(a)) and built 900 samples by randomly introducing noise. Since the subspace clustering and single-view clustering approaches have proven to be not applicable for the multi-view scenario, we applied only the multi-view clustering approaches in this experiment. We provide this dataset on our website.

(a) Samples of the stick figures data

(b) SMVC with 0 constraints



(c) SMVC with 100 constraints

Algorithm	E4FC
SMVC 0 constraints	0.700
SMVC 100 constraints	**1**
MVGen	0.760
Alt. Clus.	0.585
Multi-View 1	0.735
Multi-View 2	0.781

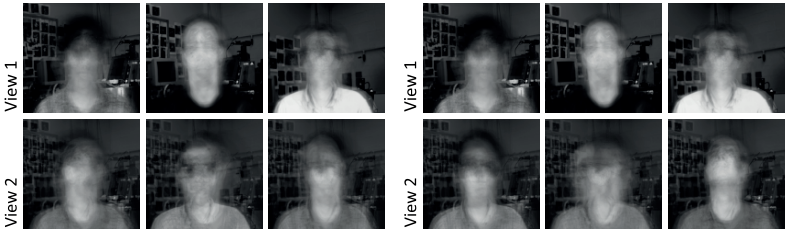
(d) Multi-view algorithm results

Figure 8.10: Evaluation of multi-view algorithms on the stick figures dataset

Although this data does not seem to be very complex, all approaches are challenged in identifying two meaningful views as shown by their clustering results (cf. Figure 8.10(d)). Even the initial result of our SMVC approach is not convincing as it produces the clustering depicted in Figure 8.10(b), which is very similar to those of the other approaches. The illustrated images correspond to the means of each detected cluster. In contrast, if we provide SMVC with 100 must-link constraints, it is able to perfectly identify the two clustering views as depicted in Figure 8.10(c). These two views differentiate between the stick figures' top position (view 1) and their leg position (view 2). Please note that we only choose 100 random constraints out of the $269{,}100$ $(= 2 \cdot (3 \cdot \binom{300}{2}))$ possible constraints. By exploiting this small amount of prior knowledge, our SMVC approach clearly outperforms all competing methods.

Case study C To show that the findings of the stick figures data also apply to more complex scenarios, we next analyze the clustering result of all multi-view approaches on the CMUFace data. This data is interesting for multi-view clustering since it consists of images taken from persons showing varying characteristics such as their facial expressions (neutral, happy, sad, angry), head positions (left, right, straight, up), and eye states (open, sunglasses). As also done in [DB10b], we randomly select 3 persons with all their images and applied PCA retaining at least 90% of the data's variance as a pre-processing.

The result of SMVC without prior knowledge for two views each with three clusters is illustrated in Figure 8.11(a). The images correspond, again, to the clusters' means. By visual inspection, we can easily identify that the first view partitions the images based on the 3 different persons. The second view, in contrast, cannot be explained easily.



(a) SMVC result with 0 constraints

(b) SMVC with 100 constraints

Algorithm	E4FC
SMVC 0 constraints	0.691
SMVC 100 constraints	**0.780**
MVGen	0.720
Alt. Clus.	0.667
Multi-View 1	0.623
Multi-View 2	0.666

(c) Multi-view algorithm results

Figure 8.11: Evaluation of multi-view clustering algorithms on the faces data

If we provide 100 constraints in order to find one view for partitioning w.r.t. the persons and another view to partition w.r.t. the head position (in total 2,592 ($= 3 \cdot \binom{32}{2} + 4 \cdot \binom{24}{2}$) possible constraints), SMVC gets the result depicted in Figure 8.11(b). Here, we can easily identify the different head positions straight, side (left and right), and up (note that we have four head positions but only search for 3 clusters). Using the original labels provided by the dataset as ground truth, i.e., the groupings based on the different persons and the grouping based on different head positions, we obtain the clustering results of Figure 8.11(c). We can see, that the unsupervised multi-view approaches all yield similar clustering qualities. They were only able to identify the first view. For SMVC, we can observe a noticeable quality improvement if we integrate prior knowledge into the clustering process.

Overall, our experiments show that SMVC is able to detect the multi-view clustering structure on a variety of datasets. It successfully solves the challenge to learn the assignment of user constraints to views such that it is able to improve its clustering results based on this prior knowledge.

8.6 Conclusion

We have presented the semi-supervised clustering method SMVC, that detects multiple clustering solutions in subspace projections and that exploits prior knowledge by incorporating instance level constraints. Our method is based on a sound Bayesian framework which models the data via multiple mixture distributions. The model uses the instance level constraints to guide the clustering of objects, and it automatically determines which views are responsible for which constraints. For learning the clustering, we use the principle of variational inference. Our experimental study has shown the high potential of SMVC to detect multiple clustering views and its capability to use the prior knowledge for improving the clustering results.

Part IV

Constraint-Based Alternative Clustering in Subspace Projections

We seldom think of what we have, but always of what we lack.
Arthur Schopenahauer

9

Introduction to Alternative Clustering

9.1 Motivation and Challenges

While the previous Part III focused on the simultaneous generation of diverse clusterings, this part will present approaches which iteratively detect new clusterings. Thereby, the knowledge of previous clusterings is used to steer the algorithm towards a novel clustering that is highly deviating from the previous clusterings but, at the same time, is of comparable quality. Since not only the data, but also the information of previous clusterings is used, approaches of this category are related to the research area of semi-supervised clustering, where additional information is supposed to guide the clustering process.

If the number of views, i.e., clustering alternatives, is known in advance, algorithms that search for all clusterings simultaneously have the advantage that the information of all clusterings can help to refine the alternatives. As a consequence all clusterings are influencing each other. In contrast, for algorithms that iteratively generate new clusterings, clustering results are only influenced by those alternatives that are detected before, but not vice versa. Especially for the semi-supervised approach SMVC (Chapter 8), a simultaneous detection of all alternatives is necessary in order to assign constraints to the correct view.

The advantage of algorithms that iteratively produce clustering alternatives is that the number of views does not necessarily have to be known in advance. Contrarily, the presented simultaneously working approaches require the number of views as an input, which clearly limits their application since for most data the number of hidden views is likely unknown and part of the knowledge discovery process. Furthermore, for many scenarios a certain knowledge base where experts have labeled or categorized the data is already available. Instead of a good algorithm for rediscovering this known information, the question whether alternative categorizations are possible is of greater interest. To answer this question it is necessary to incorporate the given knowledge into the clustering process.

In the following, we will discuss the main challenges regarding approaches that iteratively generate alternative clusterings:

- *Multiple alternatives:* Since for some applications with complex data it is possible that more than just two alternative clusterings are hidden, it should not only be possible to generate more than one alternative for one given clustering but also to incorporate more than one clustering as previous knowledge into the clustering process.
- *Global diversity:* For each iteration, the newly generated clustering should provide new insights into the data and, therefore, should deviate from *all* previously found clusterings.
- *Quality:* For each iteration, the newly generated clustering should provide valuable information and therefore should be of high clustering quality.
- *Termination:* Since for most data it is unknown how many alternative clusterings are hidden in the data, we need a certain indication factor identifying whether further alternative clusterings of high quality can be expected in the data.
- *Error tolerance:* Although previously detected clusterings are supposed to guide and influence the clustering process, it is desirable that mistakes of previous clusterings do not prevent the detection of valuable alternatives in subsequent iterations.
- *Semantic interpretability:* The discovered alternatives should not represent random object regroupings but should allow for a semantic interpretation by domain experts.

The last criterion of semantic interpretability complements the demand for a high quality of the clustering alternatives. Depending on the applied quality criterion, the presented solutions may seem arbitrary and are lacking the possibility for a semantic reasoning of experts.

We want to tackle all these challenges by a framework for alternative clustering that is based on instance level must-link constraints and searches for alternatives in subspace projections. Since pairwise instance level constraints can be modeled as a graph structure, the framework is oriented towards graph clustering methods. Before we present further details of the framework, we will discuss the existing related work in the area of iterative alternative clustering methods w.r.t. the above proposed challenges.

9.2 Related Work

In Chapter 2, the different approaches for alternative clustering have been introduced in detail. In this section, we want to present a categorization of those alternative clustering approaches that iteratively produce new clusterings based on the solutions discovered before. We will not go into algorithmic details but instead discuss the advantages and disadvantages w.r.t. the six challenges described in the previous section.

Approaches working in the original data space: The first approaches proposed in the literature focus on finding just one alternative to a given clustering [BB06, CT02, GH03, GH04, GH05, DB10a, BBD10]. They usually try to realize a trade-off between the clustering quality of the new clustering and its dissimilarity to the already known clustering. As a naive extension towards multiple alternatives, one could use the newly generated alternative as input for the same algorithm and hope to find a second alternative. But since there is no mechanism enforcing the dissimilarity to the originally provided clustering, it is very likely that the second alternative is highly similar to this first clustering. Thus, these approaches do not fulfill the first challenge of finding multiple alternative clusterings. A more promising extension of this trade-off based principle has been presented in later publications [GVG05, VE10, DB13a], where the combined dissimilarity of the new clustering to all previous clusterings is integrated into the objective function.

The fundamental problem of all approaches that are restricted to search for alternatives only in the original data space is that the number of meaningful alternatives is very limited. If just this single data representation is considered, the fixed proximity of the objects does not allow for a good diversity of the generated alternatives. Furthermore, these approaches have to rely on a trade-off between clustering quality and dissimilarity of the alternatives, which implies that the quality of the generated clusterings decreases with each iteration. Approaches of this first category are lacking a proper indication factor for stopping the search for further alternatives. Depending on the trade-off parameters, the only possible indication that no further alternatives are to be expected is an unsatisfying quality of the last found alternative or the high similarity of the last found alternative to one of the previous clusterings. Misled clusterings as input do not pose a severe problem to these algorithms since producing a high quality clustering that deviates from a bad one is not conflicting. Thus, the mentioned

approaches of this category are error tolerant. The last criterion is satisfied as well. Since all clusters are discovered in the original data space, experts should be able to discover the semantic interpretation w.r.t. the data distribution in the original space if the clustering quality is not degenerated.

In summary, approaches working only in the full-space, so far, only manage to properly tackle three of the introduced six challenges (cf. Table 9.1).

Approaches working with orthogonal space transformations: Approaches of this second category do not search just in the original data space but iteratively transform and cluster the data [DQ08, DB13b, QD09, CFD07]. The transformation of the data, which is supposed to highlight novel structures, is learned based on the clustering structure of a previous result. For the transformed data space any clustering method can be applied to achieve an alternative clustering.

Although some of the methods are presented mainly for just one alternative clustering [DQ08, QD09], they can naively be extended by proceeding each iteration based on the transformed data and the new clustering of the previous iteration like [CFD07]. However, this way, previous clusterings are only taken into account implicitly and it cannot be guaranteed that the next transformation will not produce a data space similar to one that has already been clustered. Only [DB13b] presents two approaches where multiple clusterings can directly be taken into account for the data transformation.

While techniques of the first category explicitly model the dissimilarity of new alternatives to the given clusterings in their objective function, approaches of the second category only implicitly account for the dissimilarity through transforming the data space ”orthogonally” to the given clustering structure. The dissimilarity of the clustering in the transformed space to the previous clustering is, however, not ensured. And especially for the approaches [DQ08, QD09, CFD07], a global diversity of all clusterings is uncertain since only one clustering can be taken into account for the transformation.

The quality of the alternative clusterings is solely realized by the applied clustering technique in the transformed space. The transformation itself is focused on generating a data space where a new clustering structure can be expected.

All algorithms lack a useful termination indicator. Similar to the first category's approaches, the generation of alternatives that are too similar to already discovered solutions is a sign for termination. A bad quality of a discovered alternative, however, is not a good indicator, since the next transformation might produce a space revealing a high quality clustering. Except for one of the ap-

	multiple alternatives	global diversity	high quality	termination	error tolerance	semantic interpretation
original data space	–/✓	–/○	○	–/○	✓	✓
transformed data space	–/✓	–	○	–/✓	–/○	–

Table 9.1: Overview of how well the approaches of two presented categories treat the six challenges

proaches in [CFD07], which reduces the space's dimensionality in each iteration, all approaches need to be terminated by the user.

Since in [CFD07] each transformation is applied to the transformed space of the previous iterations, a misleading transformation based on a bad clustering cannot be reversed later. Thus, mistakes also affect later iterations for this approach and might prevent the discovery of valuable alternatives. Contrarily, for both approaches of [DB13b], the transformation is applied to the original space in each iteration and depends on the input clusterings. Although it has not been examined, a bad input clustering might negatively influence the transformation w.r.t. the resulting clustering.

Although the approaches try to avoid a complete distortion of the original data by focusing on linear data transformations, the interpretability of the results in the transformed spaces is often limited and becomes especially difficult after multiple iterations of transformations.

Summarizing the above discussion, also approaches that search in space transformations for new clustering alternatives do not account for at least three of the proposed six challenges (cf. Table 9.1).

9.3 Idea of a Graph-Based Framework

In this part, we want to present a general framework that tackles all six challenges to iteratively generate alternative clusterings. The new framework will not fall into one of the categories discussed before, since it neither is restricted to the original space, nor is it transforming the data space just based on the known clusterings. As for the previously presented approaches OSCLU, ASCLU, MVGen, and SMVC, the key idea for this framework is, again, to search for alternative clusterings in subspace projections. For the reason of a better interpretability,

we will, again, focus on axis-parallel subspaces instead of general linear space transformations as do the approaches in [DQ08, DB13b, QD09, CFD07]. Unlike the approaches [DQ08, DB13b, QD09, CFD07], we do not want to determine a space transformation solely based on the given clusterings but will also take into account the clustering structure in this new space. Furthermore, we will not rely just on this new data representation for finding a new clustering but will actively incorporate the previous clustering as constraints into the clustering process.

Problem Definition 9.1 *Alternative Subspace Clustering Problem*
Given a set of objects $O = \{o_1, \ldots, o_n\} \subseteq \mathbb{R}^{|Dim|}$ with Dim being a set of dimensions and a set of known clusterings $Known = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$, find a subspace clustering $\mathcal{C} \subseteq 2^O \times 2^{Dim}$ that is dissimilar to all clusterings $\mathcal{C}_i \in Known$ and whose clusters have a high clustering quality in their respective subspaces.

Technically, we realize this by integrating each given clustering $\mathcal{C}_i \in Known$ through pairwise instance level constraints into the clustering. While the COALA approach [BB06] focuses on cannot-link constraints, which define that two objects should not be grouped together again in the next clusterings, we will use must-link, or in our case of soft constraints more precisely named as *should-link* constraints, which define that two objects should be assigned to the same cluster:

Definition 9.1 *Should-Link Constraints*
Given a set of objects O and a set of known clusterings $Known = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$, with clusterings $\mathcal{C}_i = \{C_1, \ldots, C_{k_i}\}$ and clusters $C_i \subseteq O$, the set of should-link constraints is defined as the set $ShouldLinks \subseteq O \times O$ where

$$(o_i, o_j) \in ShouldLinks \iff \forall \mathcal{C} \in Known : \neg\exists C \in \mathcal{C} : o_i \in C \wedge o_j \in C$$

Only object pairs that have not yet been grouped together in one of the known clusterings are assigned towards the set of should-link constraints. The idea is then to find a new clustering of high quality that fulfills as many should-link constraints as possible. Such a clustering will group many objects together that have not yet been grouped together.

Problem Definition 9.2 *Alternative Subspace Clustering based on Should-Links*
Given a set of objects $O = \{o_1, \ldots, o_n\} \subseteq \mathbb{R}^{|Dim|}$ with Dim being a set of dimensions and a set of should-links $ShouldLinks \subseteq O \times O$, find a subspace clustering $\mathcal{C} \subseteq 2^O \times 2^{Dim}$ that fulfills as many should-link constraints as possible, i.e., $\left|\bigcup_{(O,S)\in\mathcal{C}} (O \times O \cap ShouldLinks)\right|$ should be high, and whose clusters have a high clustering quality in their respective subspaces.

Such pairwise constraints for objects can be interpreted as graph structure. By enriching this relational graph information with the vector information of the original data, e.g., by vertex labels, we can formulate the alternative clustering problem as graph clustering task, which allows to use approaches of the popular and wide research area of graph mining.

Definition 9.2 *Vertex Labeled Graph*
Given a set of objects $O = \{o_1, \ldots, o_n\}$ and a set of should-links $ShouldLinks \subseteq O \times O$, with $o_i \in \mathbb{R}^{|Dim|}$ and Dim being a set of dimensions with $|Dim| \in \mathbb{N}_{>0}$, then the according vertex labeled graph is defined as a triple $G = (V, E, f_V)$ with a set of n vertices $V = \{v_1, \ldots, v_n\}$, a set of edges $E \subseteq M := V \times V$, and a function $f_V : V \to \mathbb{R}^{|Dim|}$ such that:

$$(v_i, v_j) \in E \iff (o_i, o_j) \in ShouldLinks \quad \wedge \quad \forall v_i \in V : f_V(v_i) = o_i$$

In order to find a good alternative clustering of the data O w.r.t. the given clusterings $Known$, the task would be to find a grouping of the graph's vertices which achieves a good quality regarding the vertices' label structure and at the same time achieves a good grouping regarding the graph structure. While for the first requirement, we are focusing on traditional subspace clustering criteria, the second requirement will be tackled by traditional graph clustering methods. Although there is no universal definition of what constitutes a good clustering of vertices within one graph, the unifying idea is that the vertices should be densely connected within each group but only sparsely connected between different groups. Like the other approaches for alternative clustering, we will concentrate on the problem of finding partitioning clusterings in the following.

Problem Definition 9.3 *Alternative Graph-Based Subspace Clustering*
Given a vertex labeled graph $G = (V, E, f_V)$ that is inferred from a set of objects $O = \{o_1, \ldots, o_n\} \subseteq \mathbb{R}^{|Dim|}$ (Dim being the set of dimensions) and a set of given clusterings $Known = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$ via the according should-link constraints, then find a clustering $\mathcal{C} \subseteq 2^V \times 2^{Dim}$ such that

- *$\mathcal{C}$ is a partitioning of V: $\forall C_i, C_j \in \mathcal{C}, C_i \neq C_j : V_i \cap V_j = \{\} \quad \wedge \quad \bigcup_{C_i \in \mathcal{C}} V_i = V$*
- *All clusters $C_i \in \mathcal{C}$ are of high quality w.r.t. their feature labels and their respective subspaces in the feature space*
- *For all clusters $C_i \in \mathcal{C}$ the grouped vertices are densely connected to each other but only weakly connected to the vertices of other clusters $C_j \in \mathcal{C}$ with $C_j \neq C_i$*

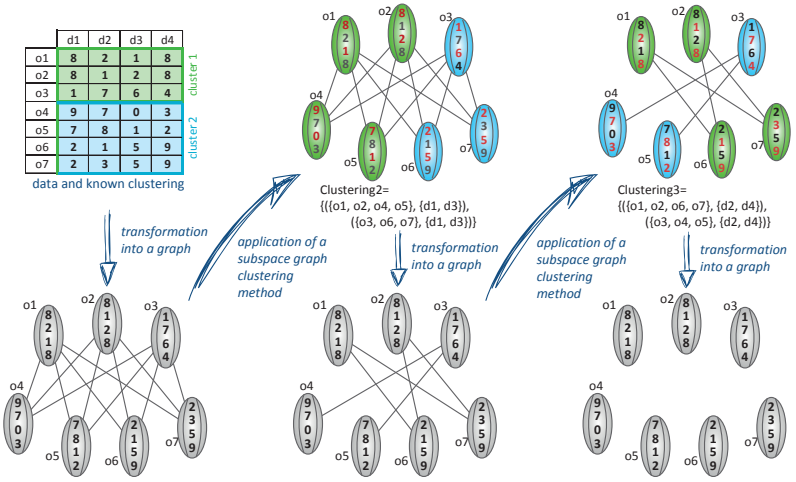


Figure 9.1: Workflow for alternative clustering with graph clustering techniques

Given the above Problem Definition 9.3, the plausible idea for the iterative computation of alternative clusterings is to transform all available information of existing clusterings into a vertex labeled graph structure and to apply a graph clustering method that is able to partition the vertices such that the both remaining requirements are fulfilled: the vertices of each partition are densely connected as well as similar w.r.t. a subset of their feature values. After each iteration the edges of the graph are updated w.r.t. the newly discovered clustering, such that the graph becomes more sparse after each iteration.

The general workflow of this iterative framework is depicted in the example of Fig. 9.1. Here, we start with just one given clustering which is transformed into the relational information of the graph through should-link constraints. Since we only know one clustering in advance, the graph is very densely connected and provides good potential to find good alternative clusterings. On this graph, we apply a graph clustering technique that is able to detect densely connected groups of vertices which show similar feature values for a subset of attributes. Although in this simple toy example the subspaces for the two discovered clusters of the new clustering are identical, our framework is not restricted to find a global relevant subspace for each clustering, which distinguishes it further from the approaches [DQ08, DB13b, QD09, CFD07]. Given the new subspace clustering result provided by the graph partitioning algorithm, we update the graph infor-

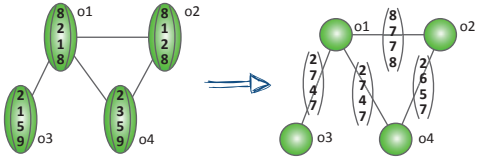


Figure 9.2: Example of transforming multi-dimensional vertex labels into multi-dimensional edge weights

mation accordingly and delete all edges (should-links) between objects that now have been clustered together: $\{(o_1, o_4), (o_1, o_5), (o_2, o_4), (o_2, o_5), (o_3, o_6), (o_3, o_7)\}$. Based on the resulting new graph, we can iteratively repeat this procedure until either the graph is too sparse to guarantee a certain minimal density of the groups or the graph has been decomposed into too many connected components to find a desired number of clusters. In our example of Fig. 9.1, after the second iteration, the pruning of the new clustering information already leads to a graph without any edges. No edges indicate that all object pairs have already been clustered together in one of the previous clusterings and we can, thus, not expect to find a highly deviating grouping anymore. Of course, this just holds for our toy example due to the small example size and we cannot expect such an ideal termination indicator of a graph without any edges to be the usual case.

Comparing the two generated alternative clusterings and the a priori given clustering, we see that they indeed cluster different sets of objects/vertices. In their respective graphs the vertices of each cluster are densely connected, already indicating the novel clustering structure. But not only the object groupings differ, also the relevant subspaces of the found alternative clusterings are different, providing new insights into the data. While the first alternative's cluster show a good clustering quality for features d_1 and d_3, the graph structure in the second iteration steered the graph clustering algorithm towards the relevant features d_2 and d_4 to find a clustering of high quality. Looking for an alternative clustering structure in a fixed data space is not very promising. Instead different feature subspaces can highlight different clustering solutions. Furthermore, the discovered relevant features highlight the main characteristics for this clustering and present a first indication for the semantic background of the object groupings.

Although the described framework and the workflow depicted in Fig. 9.1 just mention graphs with vertex labels, algorithms partitioning graphs with multi-dimensional edge weights can also be applied. In this case, the feature information of the objects that are represented by vertices just have to be transformed

into similarity vectors for the edges. A simple example for such a transformation is depicted in Fig. 9.2, where for two vertices v_i, v_j and each feature dimension d, a similarity value is calculated as the difference of the maximal possible distance and the absolute difference of the two vertices in that dimension: $similarity(v_i, v_j, d) = dist_{max} - |v_i[d] - v_j[d]|$ where we set $dist_{max} = 8$.

We will now revisit the six main challenges for approaches that iteratively generate alternative clusterings and discuss how our new graph-based framework helps to tackle each of them:

- *Multiple alternatives:* Through the flexible graph encoding of should-link constraints, our framework is able to integrate the knowledge of multiple given clusterings simultaneously. Furthermore, the iterative framework allows to generate multiple alternative clusterings, each based on the knowledge of the previously discovered solutions.
- *Global diversity:* Since the clustered vertices should exhibit a high density regarding the edges, which corresponds to a high degree of fulfillment of the should-link constraints, the newly generated clustering deviates to all previously found clusterings as it groups new objects together.
- *Quality:* The graph clustering method is a crucial part of this framework as it accounts for the quality of the novel clusterings. Besides the connectedness of the clusters' vertices, it also has to ensure their similarity in a subspace of the feature space. The latter of the two requirements ensures a high quality of the newly generated alternative clusterings.
- *Termination:* The sparsity of the generated graph, as well as the number of connected components provide a good indication regarding the potential of finding novel clustering structures.
- *Error tolerance:* Unlike other approaches, our framework does not iteratively transform the original data such that the original information is irreversibly lost. Instead, each iteration will have the feature information of the original data together with previous clusterings as side information. If bad clusterings are used to prune the graph's edges, i.e., remove should-link constraints, this can even help steering the clustering towards good solutions instead. If some of the removed edges belong to high-quality alternatives, this will still not prevent their discovery since graph clustering techniques usually do not enforce clustered vertices to be completely connected (i.e., they do not solely find cliques). The absence of just a few of the valuable edges will still allow to discover proper alternative clusterings.

- *Semantic interpretability:* Since, unlike the other transformation based approaches, we work with axis-parallel subspaces, a semantic interpretation of the results is much more intuitive. For each generated alternative, each cluster is associated with a relevant subspace. This specific set of relevant characteristics for each cluster eases a semantic reasoning.

Key to the presented framework is a properly working graph clustering algorithm, that is able to discover densely connected groups and accounts for the principles of subspace clustering for either multi-dimensional vertex labels or multi-dimensional edge weights. The clustering quality as well as the global diversity of the generated alternative clusterings only depends on the performance of the chosen graph clustering method (besides the data itself, of course). Only if the clusters are densely connected and show a good similarity structure w.r.t. a certain feature subset, we have found a valuable alternative. Therefore, the choice of the graph clustering method is very crucial.

With the increasing popularity and availability of network data over the last years, its analysis has gained much attention. The task of *graph clustering*, or more precisely the task of community detection in networks [For10], is an established mining technique and of interest for the analysis of, e.g., social networks, sensor networks, gene interaction networks, or the web. As for traditional clustering of vector data, the goal of graph clustering is to group similar vertices. Among the multiple cluster definitions, the common objective is to group the vertices into clusters such that many edges are present *within* each cluster but relatively few edges are existent *between* different clusters.

Unfortunately, the literature provides only few graph clustering approaches that are able to deal with multi-dimensional vertex labels or with multi-dimensional edge weights. Consequentially, approaches that additionally tackle the problems arising with the curse of dimensionality and noisy feature values are very rare. Existing partitioning approaches in this research area do not satisfyingly account for the problems of locally relevant feature subspaces for each cluster but instead deal only with global subspace selections or deal with the subspace determination only as post-processing step. In the following two chapters, we will present two approaches that extend the most widely used graph clustering paradigms for the problem of subspace clustering. Chapter 10 will present an according extension for spectral graph-clustering methods which is based on vertex labels. In Chapter 11, we will extend the famous modularity measure that is based on edge weights for the subspace clustering problem.

10

Spectral Subspace Clustering for Graphs with Feature Vectors

THE goal of clustering graphs annotated with feature vectors is to detect groups of vertices that are densely connected in the graph as well as similar with respect to their feature values. While early approaches treated all dimensions of the feature space as equally important, more advanced techniques consider the varying relevance of dimensions for different groups.

In this chapter, we propose a novel clustering method for graphs with feature vectors based on the principle of spectral clustering. Following the idea of subspace clustering, our method detects for each cluster an individual set of relevant features. Since spectral clustering is based on the eigendecomposition of the affinity matrix, which strongly depends on the choice of features, our method simultaneously learns the grouping of vertices *and* the affinity matrix. To tackle the fundamental challenge of comparing the clustering structures for different feature subsets, we define an objective function that is unbiased regarding the number of relevant features.

10.1 Introduction

Besides vector data, which has been the focus in the previous parts, numerous applications nowadays produce or handle network data. Besides the mere structural information, in many domains additional information for the objects is available (e.g., feature vectors annotated to the vertices). In a social network, for example, the relationships among people as well as the peoples' individual characteristics such as age or occupation might be given. Analyzing these enriched graphs regarding feature similarity *and* regarding the topological structure is challenging, though, has shown to substantially enhance clustering results [GEG+08].

The particular difficulty of clustering graphs with multi-dimensional vertex labels is that some of the features associated to the vertices might not support or even disagree with the clustering structure. It can be very futile to accommodate, e.g., private music preferences instead of research preferences to the relations shown in a co-authorship network. As known for traditional clustering of vector data, the presence of such noisy or irrelevant features is able to mask the underlying clustering structure. A solution to this problem is the paradigm of subspace clustering, which identifies clusters only in the context of their relevant features. Among the graph clustering methods that analyze graphs with feature vectors, only few methods account for irrelevant features [ZCY09, MCRE09, GFBS10, GBS12b]. Their experimental evaluation has shown that not necessarily all features exhibit a (strong) correlation with the network and that these non-correlating features can hinder a proper cluster identification.

We want to propose a novel clustering method that tackles the problem of irrelevant attributes when clustering graphs with feature vectors by extending the principle of *spectral clustering*. Spectral clustering is an established and widely used clustering paradigm which exploits the ideas of (normalized) graph cuts [vL07]. It is applicable to graph data as well as to vector data and it enjoys great popularity. For spectral clustering, the k eigenvectors belonging to the k smallest eigenvalues of, e.g., the graph's normalized Laplacian matrix, are used as cluster indicator vectors. In this setting the feature similarity is incorporated into the clustering process only as a weighting for the graph's edges. Bach and Jordan [BJ06] already pointed out that the choice of the similarity metric strongly influences the success of spectral clustering. They furthermore have shown that the presence of irrelevant features has a high impact on the clustering quality and propose a (semi-)supervised approach to learn a proper affinity matrix.

In this chapter, we present a fully unsupervised approach for clustering graphs with feature vectors. Our method simultaneously learns the grouping of vertices as well as the affinity matrix used for spectral clustering. We follow the principle of subspace clustering where for each cluster an *individual* set of features might be relevant. Thus, our method excludes locally irrelevant features which hinder the detection of good clustering results. Our contributions are:

- We present a solution for adapting spectral clustering to the problem of subspace clustering for graphs with feature vectors; for each cluster an individual set of relevant features is detected.
- We propose a computation of our objective function that is unbiased w.r.t. the number of relevant features.
- We develop the algorithm SSCG solving our objective.

10.2 Related Work

Our approach tackles the problem of clustering graph/network data in the meaning of finding homogeneous sets of vertices in a single graph. This task is also known as community detection or dense subgraph mining [Sch07].

Clustering of graphs with feature vectors. While traditional graph clustering methods concentrate on the mere structural information, there is an increasing interest in also considering complementary information such as vertex features for the clustering process. Methods for this problem setting generally assume that clusters based on the graph's topology and the vertices' features are more meaningful than those based only on one characteristic. In [HZZL02] structural and feature information are combined into a single distance function, which can result in clusters with neither a specific graph nor a specific feature pattern. [GEG+08] tackles the problem from the features' perspective and extends the k-center problem by an internal connectedness constraint. The authors of [STM07] use a normalized modularity definition where vertex features are incorporated as edge weights. For minimizing this normalized modularity, a spectral clustering approach is used. [ATMF12] introduces a parameter-free approach following the idea of compression. All the above approaches are not able to detect similarities among vertices based on feature subsets. Similar to traditional full-space clustering for vector data, their results will become less meaningful in the presence of irrelevant features. For vector data, the paradigm of subspace clustering [KKZ09, AWY+99] reduces the influence of irrelevant features.

Varying relevance of dimensions. Only few approaches were presented so far that attend to the clustering of feature labeled graphs from a subspace clustering perspective. The approach of [ZCY09] defines a feature augmented graph, where features are modeled as additional vertices linked to those original vertices showing the specific value for this feature. For the final clusters, objects are only pairwise similar and no particular relevant features can be defined for each cluster. The principle of detecting an individual subset of relevant dimensions for each cluster is, so far, only fulfilled by three approaches [MCRE09, GFBS10, GBS12b]. While [MCRE09, GFBS10] exploit the notion of quasi-cliques, which poses strong restrictions regarding the clusters' feature range and their diameter, the method of [GBS12b] follows a density based cluster notion for subgraphs as well as for the feature space. The work of [MCRE09] generates a huge amount of overlapping clusters leading to high redundancy in the clustering result. To control the level of redundancy, [GFBS10, GBS12b] propose models for redundancy handling, introducing additional parameters the user has to specify. Our approach based on spectral clustering determines a partitioning of the vertices and, thus, does not suffer from redundancy.

Spectral clustering. Spectral clustering [vL07] is suitable for vector data [YHJ09, BJ06] and graph data [STM07]. Even though the strong influence of the affinity matrix on the clustering result has already been noted, so far, no approach exists that considers the affinity matrix as part of the unsupervised learning process. [BJ06] confirms the detrimental effect of irrelevant features for spectral clustering. They provide a method for learning the affinity matrix in a (semi-)supervised fashion. Besides assuming a given partition, [BJ06] determines a *global weighting* of the features. As shown for vector data, it might be the case that for different clusters different features prove to be irrelevant. In such cases no feature subset will help to uncover all clusters, which makes it desirable to find clusters with *individual sets of relevant features*. Our method simultaneously learns the partition and the underlying affinity matrix, where one important goal is to diminish the influence of irrelevant features for each partition individually.

10.3 Model

In this section, we present our model to cluster feature labeled graphs in subspace projections. The input for our method is a vertex labeled graph $G = (V, E, l)$ with

vertices $V = \{1, \ldots, N\}$, edges $E \subseteq V \times V$, and a labeling function $l : V \to \mathbb{R}^D$, where $Dim = \{1, \ldots, D\}$ is the set of dimensions. We assume normalized feature vectors in the range $[0, 1]$ per dimension.

10.3.1 Preliminaries

Among the multitude of different cut definitions, we focus on the frequently used normalized cut because of its strength in avoiding unbalanced cuts [SM00]. Intuitively, the goal of clustering based on normalized cuts is to find a K-partitioning of the nodes that minimizes the inter-cluster connectivity while at the same time maximizing the intra-cluster connectivity. For this purpose, let $\mathbb{P}^{N,K}$ be the set of all possible (complete and disjoint) K-way partitionings of the set V. It can be represented by the set of binary matrices:

$$\mathbb{P}^{N,K} := \left\{ \mathbf{A} \in \{0,1\}^{N \times K} \mid \sum_{k=1}^{K} \mathbf{a}_k = \mathbf{1}_N \wedge \forall k : \mathbf{a}_k \neq \mathbf{0}_N \right\}$$

where $\mathbf{a}_k$ is the k-th column of the binary matrix $\mathbf{A}$, $\mathbf{1}_N$ is the vector containing only entries equal to 1, and $\mathbf{0}_N$ contains only entries equal to 0. Each $\mathbf{A} \in \mathbb{P}^{N,K}$ represents one possible partitioning and each column vector $\mathbf{a}_k$ stands for one specific group of this partitioning. This definition automatically ensures the orthogonality of the vectors $\mathbf{a}_k$. We denote by $V(\mathbf{a}_k)$ the vertices belonging to group k.

Let $\mathbf{W} = (w_{u,v})_{u,v=1}^{N}$ be the adjacency matrix of a graph G. The cut-value between two groups $V(\mathbf{a}_k)$ and $V(\mathbf{a}_{k'})$ is defined as:

$$cut_{\mathbf{W}}(V(\mathbf{a}_k), V(\mathbf{a}_{k'})) = \sum_{u \in V(\mathbf{a}_k), v \in V(\mathbf{a}_{k'})} w_{u,v} = \mathbf{a}_k^T \cdot \mathbf{W} \cdot \mathbf{a}_{k'}$$

The goal of the normalized cut problem is to find a partitioning $\mathbf{A} \in \mathbb{P}^{N,K}$ that minimizes the following function:

$$nCut_{\mathbf{W}}(\mathbf{A}) = \sum_{k=1}^{K} \frac{\mathbf{a}_k^T \cdot \mathbf{W} \cdot (\mathbf{1}_N - \mathbf{a}_k)}{\mathbf{a}_k^T \cdot \mathbf{W} \cdot \mathbf{1}_N} \tag{10.1}$$

The numerator calculates the sum of weights over outgoing edges of cluster k, while the denominator calculates the sum of weights over internal and outgoing edges of cluster k. Thus, the normalized cut trades off low inter-cluster connectivity and high intra-clustering connectivity.

Since optimizing Equation 10.1 is intractable [SM00], spectral clustering aims at solving a relaxed problem. By substituting the binary-valued cluster indicator vectors $\mathbf{a}_k$ with real-valued vectors, the above problem transforms into a tractable eigenvector problem based on the normalized graph Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}$. Here, we use $\mathbf{D} = diag(d_1, \ldots, d_k)$ with $d_u = \sum_{v=1}^{N} w_{u,v}$. The K-way partitioning of the nodes can finally be obtained by determining the first K eigenvectors of $\mathbf{L}$, considering each of the N rows as a K-dimensional vector, and by clustering them based on, e.g., k-means. We refer to [vL07] for more details.

Integrating feature vectors using kernels. Since for the normalized cut and spectral clustering the data is represented by the matrix $\mathbf{W}$, it is easy to incorporate additional aspects into the process of clustering. For example, by applying a kernel transformation $k(x, y)$ on the feature vectors, we can enrich $\mathbf{W}$ by the similarity of these features:

$$w_{u,v} = k(\mathbf{x}, \mathbf{y}) \cdot \mathbb{I}((u, v) \in E) \tag{10.2}$$

where $\mathbf{x} = l(u)$ and $\mathbf{y} = l(v)$ are the feature vectors of vertices u and v, and $\mathbb{I}$ is the indicator function.

In the following, we focus on radial basis function kernels (RBF kernels) where the kernel value $k(\mathbf{x}, \mathbf{y})$ just depends on the norm of $\mathbf{x} - \mathbf{y}$, i.e., $k(\mathbf{x}, \mathbf{y}) = k(\|\mathbf{x} - \mathbf{y}\|)$.[1] Furthermore, in our scenario, it is natural to restrict the consideration to kernels having a non-negative derivative, i.e., $\frac{d}{dx}k(x) \geq 0$ for $x \geq 0$. Thus, increasing the 'dissimilarity' between two feature vectors, decreases the kernel value. These properties hold for a variety of kernels such as, e.g., the Gaussian, Rational quadratic, or Exponential kernel [Gen02].

10.3.2 Normalized Cut in Subspace Projections

As mentioned in the introduction, one cannot expect to find clusters in the full dimensional space but in subspace projections of the data. For example, the graph depicted in Figure 10.1 exhibits no group of vertices being similar w.r.t. all three dimensions. Thus, instead of considering the (unweighted) Euclidean norm between two feature vectors, we use the weighted Euclidean norm

$$\|\mathbf{x} - \mathbf{y}\|_{\mathbf{s}} := \sqrt{(\mathbf{x}-\mathbf{y})^T diag(\mathbf{s})(\mathbf{x}-\mathbf{y})} \text{ s.t. } \mathbf{s} \in \{[0, 1]^D \mid \sum_{i=1}^{D} s_i{=}1\}$$

[1] To simplify the notations, we will overload the function symbol k. It can either be a binary function or an unary one. The actual use will be clear from the context.

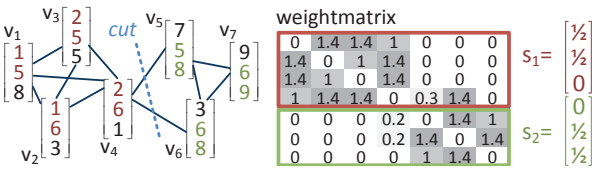


Figure 10.1: Weight matrix for subspaces s_1 and s_2 belonging to partitioning $\{\{v_1, v_2, v_3, v_4\}, \{v_5, v_6, v_7\}\}$.

Based on the subspace vector $\mathbf{s}$, we can weight the importance of individual dimensions. Noisy or uninteresting dimensions can be excluded by choosing $s_i \to 0$.

Since we do not know a priori in which subspaces the clusters are located, unlike most approaches, we cannot assume the matrix $\mathbf{W}$ to be a priori given or static anymore. *In our method, we simultaneously learn the matrix* $\mathbf{W}$ *as well as the object grouping* $\mathbf{A}$.

Let $\mathbf{s}$ be a subspace vector, the matrix $\mathbf{W}_\mathbf{s} = (\hat{w}_{u,v})_{u,v=1}^{N}$ is defined by

$$\hat{w}_{u,v} = k(\|l(u) - l(v)\|_\mathbf{s}) \cdot \mathbb{I}((u,v) \in E) \tag{10.3}$$

The matrix $\mathbf{W}_\mathbf{s}$ represents the graph when projected to a *single* subspace. Consequently, it corresponds to a global dimensionality reduction. We, however, are interested in finding locally relevant subspaces: each cluster is associated with an *individual* subspace. In our toy example of Figure 10.1, for the group $\{v_1, v_2, v_3, v_4\}$ features 1 and 2 are relevant, while for group $\{v_5, v_6, v_7\}$ features 2 and 3 are interesting. Thus, instead of considering a single subspace vector $\mathbf{s}$, we are interested in finding a matrix $\mathbf{S} \in [0,1]^{D \times K}$, where each column represents a (possibly different) subspace vector. Technically, it has to hold $\mathbf{S} \in \mathbb{S}^{D,K}$ where

$$\mathbb{S}^{D,K} := \{\mathbf{S} \in [0,1]^{D \times K} \mid \forall k : \|\mathbf{s}_k\|_1 = 1\}$$

and $\mathbf{s}_k$ denotes the k-th column of $\mathbf{S}$.

Using individual subspaces introduces a further challenge. The weights between the vertices do not solely depend on $\mathbf{S}$ but they depend on $\mathbf{A}$, too. What is the weight between two vertices u and v? In the case that both nodes belong to the k-th cluster, $u, v \in V(\mathbf{a}_k)$, it is clear that the weight is determined by $(\mathbf{W}_{\mathbf{s}_k})_{u,v}$. Though, how to handle the case where the nodes belong to different clusters? A principled answer to this question can be given by exploiting the relation between random walks and the normalized cut [MS01].

Considering the graph as a Markov Chain, the problem of minimizing the normalized cut can be interpreted as finding a partitioning such that a random walk (using the stationary distribution of the Markov Chain) stays long within the same cluster and seldom moves between clusters. More technically, in [MS01] the following is shown: Let $Pr[A \to B|A]$ denote the probability of the random walk to transition from vertex set A to vertex set B in a single step *given that the walk starts in a vertex from* A. Then, the normalized cut is equal to $\sum_{k=1}^{K} Pr[O_k \to (V \backslash O_k)|O_k]$ where O_k denotes the vertices belonging to the k-th cluster.

The *conditional* probability in the equation above provides us with the answer to our original question: Let be $u \in V(\mathbf{a}_k)$ and $v \in V(\mathbf{a}_{k'})$, the weight between vertex u and v is $(\mathbf{W}_{\mathbf{s}_k})_{u,v}$, while the weight between v and u is $(\mathbf{W}_{\mathbf{s}_{k'}})_{v,u}$. One has to condition on the subspace $\mathbf{s}_i$ where the random walk starts. The effects can be nicely observed for the weight matrix of Figure 10.1 (for simplicity, we used $k(x) = 1/x$ in the toy example). Here the i-th row contains the weights based on the subspace of the cluster the vertex v_i belongs to. Thus, in the fourth row, for example, the subspace $\mathbf{s}_1$ is used, while $\mathbf{s}_2$ is used in the fifth. As a consequence, inter-cluster edges (e.g., (v_4, v_5) and (v_4, v_6)) might have different weights in different rows. Intuitively, when measuring the 'goodness' of cluster k, we project the whole graph to the subspace $\mathbf{s}_k$ and analyze how well cluster k is separated. From the k-th cluster's point of view it does not matter how well it is separated in other subspaces $\mathbf{s}_{k'}$.

Summarizing, given the subspace matrix $\mathbf{S}$ and the object groupings $\mathbf{A}$, the weight matrix is formalized as:

Definition 10.1 *Subspace Dependent Weight Matrix*
Let $\mathbf{S} \in \mathbb{S}^{D,K}$ *be a matrix representing* K *subspace vectors and* $\mathbf{A} \in \mathbb{P}^{N,K}$ *a* K*-way partitioning. The subspace dependent weight matrix is defined as*

$$\mathbf{W}_{\mathbf{S},\mathbf{A}} = \sum_{k=1}^{K} \mathbf{W}_{\mathbf{s}_k} \circ \left(\mathbf{a}_k \cdot \mathbf{1}_N^T\right)$$

where $\mathbf{s}_k$ *(*$\mathbf{a}_k$*) is the* k*-th column of* $\mathbf{S}$ *(*$\mathbf{A}$*),* $\mathbf{W}_{\mathbf{s}_k}$ *as defined in Equation 10.3, and* $\circ$ *is the Hadamard product.*

By using the term $\mathbf{a}_k \cdot \mathbf{1}_N^T$, we ensure that the whole graph is projected to the subspace $\mathbf{s}_k$ when considering the vertices $V(\mathbf{a}_k)$. Note that in general the weight matrix $\mathbf{W}_{\mathbf{S},\mathbf{A}}$ might not be symmetric even if the underlying graph is undirected (see Figure 10.1). However, assuming an undirected graph, the matrix shows a

certain kind of (a)symmetry: each subblock of $\mathbf{W}_{\mathbf{S},\mathbf{A}}$ induced by a single cluster k, i.e., each block $\mathbf{a}_k \cdot \mathbf{a}_k^T$, is symmetric. This follows since all objects of the same cluster are projected to the same subspace. The asymmetry of $\mathbf{W}_{\mathbf{S},\mathbf{A}}$ is only caused by edges connecting different clusters.

We are now ready to formalize our overall objective. Our goal is to find a partitioning $\mathbf{A}$ and individual subspaces $\mathbf{S}$ such that the normalized cut based on the matrix $\mathbf{W}_{\mathbf{S},\mathbf{A}}$ is minimized. Thus, we aim at simultaneously optimizing multiple objectives: a) we maximize the intra-cluster connectivity and the intra-cluster similarity of the feature vectors w.r.t. the individually selected subspaces, b) we minimize the inter-cluster connectivity and inter-cluster similarity of the feature vectors in the corresponding subspaces.

Definition 10.2 *Minimum Normalized Subspace Cut*
Given the graph $G = (V, E, l)$ and the number of clusters K, the minimum normalized subspace cut (MNSC) is the problem of finding $\mathbf{S}^* \in \mathbb{S}^{D,K}$, $\mathbf{A}^* \in \mathbb{P}^{N,K}$ such that

$$(\mathbf{A}^*, \mathbf{S}^*) = \underset{\mathbf{A}\in\mathbb{P}^{N,K}, \mathbf{S}\in\mathbb{S}^{D,K}}{\arg\min} \{NSCut(\mathbf{A}, \mathbf{S})\}$$

$$\textit{where} \quad NSCut(\mathbf{A}, \mathbf{S}) := \sum_{k=1}^{K} \frac{\mathbf{a}_k^T \cdot \mathbf{W}_{\mathbf{S},\mathbf{A}} \cdot (\mathbf{1}_N - \mathbf{a}_k)}{\mathbf{a}_k^T \cdot \mathbf{W}_{\mathbf{S},\mathbf{A}} \cdot \mathbf{1}_N}$$

10.3.3 Subspace Unbiased Cut Computation

When considering the *unweighted* Euclidean norm, the distance between objects increases with increasing subspace dimensionality. Thus, comparing the cut-value in, e.g., a 1-dimensional subspace with the cut-value in a 5-dimensional subspace is not revealing at all. Since our goal, however, is to pick the best subspace among all possible subspaces, we have to realize a fair comparison of the cut values. The cut values should not be biased to specific dimensionalities of the subspaces.

While solutions to this problem have been proposed in the subspace clustering community (e.g., [AKMS07a]), in our scenario two aspects are worth mentioning: (1) We consider the *normalized* cut. Computing the fraction of cut values might appear to be unbiased; though, it is not. Consider, e.g., the case that the *second* derivative of the kernel function is negative (e.g., the Gaussian and Exponential kernel). In this case, the cut would be biased to low-dimensional clusters. The reason is that the kernel values drop quicker for small norm values than for

larger ones. Thus, informally, when adding a further dimension, the edges within a cluster (high feature similarity) loose much of their kernel value while the edges between clusters (dissimilar features) receive almost the same value. Thus, we loose discrimination power between the inter-cluster and intra-cluster edges. (2) We consider the *weighted* Euclidean norm where the weights have to sum up to 1. However, even in this case we observe an increase of the distance values with increasing dimensionality. Thus, overall, also for our objective function we have to realize an unbiased computation.

Our solution. A simple solution to avoid dimensionality bias would be to introduce a regularization parameter that controls the sparsity/density of the vectors $\mathbf{s}_k$. This frequently used principle of regularization, however, is rather ad hoc and introduces additional regularization parameters which are often hard to set. We do not want to introduce additional parameters. Instead, we extend the results known from the area of subspace clustering [AKMS07a].

The basic idea of our principle is to adapt the computation of the norm such that we obtain an unbiased estimation, i.e., the expected distance (and its variance) between the feature vectors should be constant and, thus, independent of the selected subspace. For this purpose, we first formalize:

Definition 10.3 *Unbiased parametric family*
Given a parametric family $\mathcal{F} = \{f_{\mathbf{s}} \mid \mathbf{s} \in \Theta\}$ of functions $f_{\mathbf{s}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and a (multi-dimensional) probability density function τ over $\mathbb{R}^d$. $\mathcal{F}$ is called unbiased if

$$\forall \mathbf{s}, \mathbf{s}' \in \Theta : \mathrm{E}\left[f_{\mathbf{s}}(X,Y)\right] = \mathrm{E}\left[f_{\mathbf{s}'}(X,Y)\right] < \infty$$

$$\forall \mathbf{s}, \mathbf{s}' \in \Theta : \mathrm{Var}\left[f_{\mathbf{s}}(X,Y)\right] = \mathrm{Var}\left[f_{\mathbf{s}'}(X,Y)\right] < \infty$$

where X and Y are i.i.d. with $X \sim \tau, Y \sim \tau$

The family $\mathcal{F}$ might, for example, be the set of functions computing the weighted Euclidean norm, i.e., we would have $f_{\mathbf{s}}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_{\mathbf{s}}$ where Θ consists of all valid subspace vectors. The probability density function τ corresponds to the null model the feature vectors are generated from. Intuitively, it corresponds to the distribution when expecting no clusters in the data. Based on our setting, it corresponds to the uniform distribution over the hypercube $[0,1]^D$, i.e., $\tau(\mathbf{x}) = 1$ if $\mathbf{x} \in [0,1]^D$, 0 otherwise.

If a family $\mathcal{F}$ of functions is unbiased, we can do a fair comparison between the function values of $f_{\mathbf{s}}(\mathbf{x}, \mathbf{y})$ and $f_{\mathbf{s}'}(\mathbf{x}, \mathbf{y})$. As mentioned above, the weighted Euclidean norm is not unbiased. Though, we can show the following:

Theorem 10.1 *Let Θ_D be the set of all possible D-dimensional subspace vectors and*

$$\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}} := \frac{\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}} - \mathrm{E}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]}{\sqrt{\mathrm{Var}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]}} + \min_{\mathbf{s}'\in\Theta_D} \frac{\mathrm{E}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}'}\right]}{\sqrt{\mathrm{Var}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}'}\right]}}$$

The parametric family $\mathcal{F}=\{\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}} |\ \mathbf{s}\in\Theta_D\}$ is unbiased.

Proof 10.1 *Using the abbreviations $\mu_{\mathbf{s}} := \mathrm{E}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]$ and $\sigma_{\mathbf{s}} := \sqrt{\mathrm{Var}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]}$, then for all $\mathbf{s}\in\Theta_D$ it holds*

- $\mathrm{E}\left[\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}}\right] = \frac{1}{\sigma_{\mathbf{s}}}(\mathrm{E}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right] - \mu_{\mathbf{s}}) + \min_{\mathbf{s}'\in\Theta_D} \frac{\mu_{\mathbf{s}'}}{\sigma_{\mathbf{s}'}} = \min_{\mathbf{s}'\in\Theta_D} \frac{\mu_{\mathbf{s}'}}{\sigma_{\mathbf{s}'}} = c_1$
- $\mathrm{Var}\left[\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}}\right] = \mathrm{E}\left[(\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}} - c_1)^2\right] = \mathrm{E}\left[\left(\frac{\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}-\mu_{\mathbf{s}}}{\sigma_{\mathbf{s}}}\right)^2\right] =$
  $\frac{1}{\sigma_{\mathbf{s}}^2}\cdot \mathrm{E}\left[(\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}} - \mu_{\mathbf{s}})^2\right] = \frac{1}{\sigma_{\mathbf{s}}^2}\cdot \mathrm{Var}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right] = \frac{1}{\sigma_{\mathbf{s}}^2}\cdot \sigma_{\mathbf{s}}^2 = 1 = c_2$

Since c_1 and c_2 are independent from $\mathbf{s}\in\Theta_D$, the parametric family $\mathcal{F}$ is unbiased.

Intuitively, $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$ is the z-score normalized version of $\|.\|_{\mathbf{s}}$. Thus, instead of measuring the absolute norm between two features, we measure the deviation to the expected value. Since $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$ is guaranteed to be non-negative, it is possible to replace the value of $\|.\|_{\mathbf{s}}$ in Equation 10.3 by the unbiased measure $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$. The question remains whether $k(\lfloor\!\uparrow \mathbf{x}-\mathbf{y} \uparrow\!\rfloor_{\mathbf{s}})$ still corresponds to a valid kernel transformation [Gen02]. While in general the use of $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$ does not lead to a valid Mercer kernel, we can show:

Theorem 10.2 *Given the exponential kernel $k_\theta(t) = e^{-\frac{t}{\theta}}$ with scaling parameter θ. When solving the MNSC problem, replacing $\|.\|_{\mathbf{s}}$ in Equation 10.3 by $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$ is equivalent to using the (original!) norm $\|.\|_{\mathbf{s}}$ in combination with the exponential kernel based on the scaling parameter $\theta_{\mathbf{s}} := \theta\cdot\sqrt{\mathrm{Var}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]}$.*

Proof 10.2 *Let $\widetilde{\mathbf{W}}_{\mathbf{s}}$ be the weight matrix according to Eq. 10.3 using the (unbiased but potentially invalid) kernel values $k_\theta(\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}})$, and let $\mathring{\mathbf{W}}_{\mathbf{s}}$ be the weight matrix using the (valid) kernel values $k_{\theta_{\mathbf{s}}}(\|.\|_{\mathbf{s}})$. We show that using $\widetilde{\mathbf{W}}_{\mathbf{s}}$ is equivalent to using $\mathring{\mathbf{W}}_{\mathbf{s}}$ when solving the MNSC problem.*

- *We first reformulate the objective function: Since $\mathbf{A}$ is a partitioning, for all k, k' with $k\neq k'$ it holds $\mathbf{a}_k^T\cdot\left(\mathbf{a}_{k'}\cdot\mathbf{1}_N^T\right) = \mathbf{0}_N^T$. Thus, the objective function in Def. 10.2 can be written as*

$$NSCut(\mathbf{A},\mathbf{S}) = \sum_{k=1}^{K} \frac{\mathbf{a}_k^T\cdot\left(\sum_{k'=1}^{K}\mathbf{W}_{\mathbf{s}_{k'}}\circ\left(\mathbf{a}_{k'}\cdot\mathbf{1}_N^T\right)\right)\cdot(\mathbf{1}_N-\mathbf{a}_k)}{\mathbf{a}_k^T\cdot\left(\sum_{k'=1}^{K}\mathbf{W}_{\mathbf{s}_{k'}}\circ(\mathbf{a}_{k'}\cdot\mathbf{1}_N^T)\right)\cdot\mathbf{1}_N}$$

$$= \sum_{k=1}^{K} \frac{\mathbf{a}_k^T\cdot(\mathbf{W}_{\mathbf{s}_k}\circ(\mathbf{a}_k\cdot\mathbf{1}_N^T))\cdot(\mathbf{1}_N-\mathbf{a}_k)}{\mathbf{a}_k^T\cdot(\mathbf{W}_{\mathbf{s}_k}\circ(\mathbf{a}_k\cdot\mathbf{1}_N^T))\cdot\mathbf{1}_N} = \sum_{k=1}^{K} \frac{\mathbf{a}_k^T\cdot\mathbf{W}_{\mathbf{s}_k}\cdot(\mathbf{1}_N-\mathbf{a}_k)}{\mathbf{a}_k^T\cdot\mathbf{W}_{\mathbf{s}_k}\cdot\mathbf{1}_N}$$

  Here, $\mathbf{W}_{\mathbf{s}_k}$ is used as a placeholder for either $\widetilde{\mathbf{W}}_{\mathbf{s}_k}$ or $\mathring{\mathbf{W}}_{\mathbf{s}_k}$.

- *Using the abbreviations from the proof of Theorem 10.1, the kernel functions can be reformulated as*

$$k_\theta(\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}) = e^{-\frac{\frac{\|\cdot\|_{\mathbf{s}}-\mu_{\mathbf{s}}}{\sigma_{\mathbf{s}}}+c_1}{\theta}} = e^{\frac{\mu_{\mathbf{s}}}{\sigma_{\mathbf{s}}\cdot\theta}-\frac{c_1}{\theta}} \cdot e^{-\frac{\|\cdot\|_{\mathbf{s}}}{\sigma_{\mathbf{s}}\cdot\theta}} = c_{\mathbf{s}} \cdot k_{\theta_{\mathbf{s}}}(\|.\|_{\mathbf{s}})$$

  where $c_{\mathbf{s}} := e^{\frac{\mu_{\mathbf{s}}}{\sigma_{\mathbf{s}}\cdot\theta}-\frac{c_1}{\theta}}$ *is a constant depending on the subspace* $\mathbf{s}$*. It follows:* $\widetilde{\mathbf{W}}_{\mathbf{s}} = c_{\mathbf{s}} \cdot \mathring{\mathbf{W}}_{\mathbf{s}}$
- *Plugging this result into the reformulated objective function, we get:*

$$\sum_k \frac{\mathbf{a}_k^T \cdot \widetilde{\mathbf{W}}_{\mathbf{s}_k} \cdot (\mathbf{1}_N - \mathbf{a}_k)}{\mathbf{a}_k^T \cdot \widetilde{\mathbf{W}}_{\mathbf{s}_k} \cdot \mathbf{1}_N} = \sum_k \frac{\mathbf{a}_k^T \cdot c_{\mathbf{s}_k} \cdot \mathring{\mathbf{W}}_{\mathbf{s}_k} \cdot (\mathbf{1}_N - \mathbf{a}_k)}{\mathbf{a}_k^T \cdot c_{\mathbf{s}_k} \cdot \mathring{\mathbf{W}}_{\mathbf{s}_k} \cdot \mathbf{1}_N} = \sum_k \frac{\mathbf{a}_k^T \cdot \mathring{\mathbf{W}}_{\mathbf{s}_k} \cdot (\mathbf{1}_N - \mathbf{a}_k)}{\mathbf{a}_k^T \cdot \mathring{\mathbf{W}}_{\mathbf{s}_k} \cdot \mathbf{1}_N}$$

  $\Rightarrow$ *using* $\widetilde{\mathbf{W}}_{\mathbf{s}}$ *or* $\mathring{\mathbf{W}}_{\mathbf{s}}$ *is equivalent.*

Thus, using $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$ will still lead to a valid Mercer kernel. Furthermore, to realize an unbiased computation of the cut, we can simply adapt the scaling parameter of the exponential kernel. We actually do not have to compute $\lfloor\!\uparrow . \uparrow\!\rfloor_{\mathbf{s}}$; particularly, the term $\mathrm{E}\left[\|\mathbf{x}-\mathbf{y}\|_{\mathbf{s}}\right]$ vanishes completely.

Overall, when using the exponential kernel, we can obtain a subspace unbiased cut computation while still preserving the properties of a valid Mercer kernel. Interestingly, the exponential kernel does not only show these nice theoretical properties, it also has outperformed all other kernels in our empirical studies (Figure 10.2). Unlike many methods with artificial regularization parameters, our subspace unbiased cut computation does not introduce any additional parameters.

10.3.4 Complexity Analysis

Statements like "the normalized cut problem is NP-complete" have to be regarded carefully. Since usually these problems are formulated as optimization problems, the classical definitions of complexity classes – which were designed for decision problems – cannot be applied straightforward [Kan92]. Thus, we consider the decision problem version of MNSC.

Definition 10.4 *Decision problem version of MNSC*
The decision problem version to the MNSC optimization problem is: Given a graph $G = (V, E, l)$*, the number of groups* K*, and a constant* C*. Is there a normalized subspace cut with value* $\leq C$*?*

Theorem 10.3 *The decision problem version of MNSC is* NP-complete *if the kernel function can be evaluated in polynomial time w.r.t.* G, K, *and* C.

Proof 10.3 *a) MNSC is NP-hard: We provide a polynomial reduction of the 'usual' normalized cut problem NCUT (which does not handle feature vectors) to our MNSC problem, i.e., NCUT* $\leq_P$ *MNSC.*

- *Input mapping: The input* $G{=}(V, E)$ *of NCUT is mapped to an input* $G' = (V', E', l')$ *of MNSC with* $V' = V$, $E' = E$ *and* $l'(v) = 0$ *for all* $v \in V$ *(i.e., each node has the same feature vector). This transformation can be done in polynomial time.*
- *MNSC generates a valid NCUT solution: Let* $\mathbf{X}$ *denote the adjacency matrix used for NCUT. Since all feature vectors in* G' *are identical, it holds* $\mathbf{W}_{\mathbf{S},\mathbf{A}} = k(0) \cdot \mathbf{X}$. *The constant value* $k(0)$ *does not affect the resulting optimal cut. Thus, the solution of MNSC corresponds to the solution of NCUT. Since NCUT is NP-complete [SM00], MNSC is NP-hard.*

b) MNSC is in NP: We use the verifier-based definition of NP. Given a certificate ($\mathbf{A}$,$\mathbf{S}$*), we prove that its correctness, i.e., the equation* $NSCut(\mathbf{A}, \mathbf{S}) \leq C$, *can be verified in* polynomial time. *The following complexities hold:*

- *computing the norm:* $T_n := \mathcal{O}(d)$
- *computing the kernel:* $T_k := \mathcal{O}(p)$, *with* p *is a polynomial*
- *computing* $\mathbf{W}_{\mathbf{S},\mathbf{A}}$ *(Def. 10.1):* $T_w := \mathcal{O}(|E| \cdot (T_n + T_k))$
- *computing* $NSCut$ *(Def. 10.2):* $T_c := \mathcal{O}(k \cdot |E|)$

For T_w *and* T_c *we exploited the sparsity of the weight matrix and the fact that each vertex belongs to a single cluster. Overall, computing* $NSCut(\mathbf{A}, \mathbf{S})$ *is in class* $\mathcal{O}(|E| \cdot (d + p + k)) \in$ *P. Since the verification is in P, MNSC is in NP.*

c) combining a) and b) $\Rightarrow$ *MNSC is NP-complete.*

Thus, even though our model uses an adaptive $\mathbf{W}$ depending on $\mathbf{S}$ and $\mathbf{A}$, its complexity class is identical to the one of the normalized cut where $\mathbf{W}$ is assumed to be static. However, two aspects have to be noted: First, the input for our model is more complex since we consider feature labeled graphs. Second, these results apply for the decision problem version. It does not necessarily follow that the optimization problems are equally complex, i.e., the degrees of approximability might be different [Kan92]. This study is left for future work.

10.4 Algorithm

As shown by Theorem 10.3, we cannot expect to find an efficient algorithm computing an exact solution to the MNSC problem. Alternatively, we design an algorithm computing an approximate solution based on the following observations: (1) When keeping the matrix $\mathbf{W}$ fix, determining the optimal partitioning $\mathbf{A}$ is independent of $\mathbf{S}$ and reduces to the traditional normalized cut problem. (2) As shown in the proof of Theorem 10.2, the objective function can be written as

$$\sum_{k=1}^{K} g_{\mathbf{a}_k}(\mathbf{s}_k) \quad \text{with} \quad g_{\mathbf{a}}(\mathbf{s}) := \frac{\mathbf{a}^T \cdot \mathbf{W}_{\mathbf{s}} \cdot (\mathbf{1}_N - \mathbf{a})}{\mathbf{a}^T \cdot \mathbf{W}_{\mathbf{s}} \cdot \mathbf{1}_N}$$

Thus, if the matrix $\mathbf{A}$ is given, the subspaces $\mathbf{s}_k$ can be optimized for each cluster *independently*. This independence drastically reduces the hardness. Since for given $\mathbf{A}$ and $\mathbf{S}$, the matrix $\mathbf{W}$ is completely determined, these observations naturally lead to an iterative algorithm where we optimize one variable while keeping the others fix. Such a procedure of alternating optimization is well established for many tasks. Our method works as follows:

1: initialize $\mathbf{W}^{(0)}$
2: for($t = 1, \ldots$)
3: compute normalized Laplacian $\mathbf{L}^{(t)} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}^{(t-1)}$
4: compute first k eigenvectors $u_1, \ldots, u_k$ of $\mathbf{L}^{(t)}$
5: determine $\mathbf{A}^{(t)}$ by performing k-means clustering on $U = [u_1, \ldots, u_k] \in \mathbb{R}^{N \times k}$
6: determine $\mathbf{S}^{(t)} = [\mathbf{s}_1, \ldots, \mathbf{s}_k]$ by minimizing $g_{\mathbf{a}_i}(\mathbf{s}_i)$
(can be done for each cluster i independently)
7: compute $\mathbf{W}^{(t)}$ (cf. Def. 10.1) based on $\mathbf{A}^{(t)}$ and $\mathbf{S}^{(t)}$
8: stop if cut-value has converged

We continue by briefly discussing each step of the method.

Initialization & update of $\mathbf{W}$ Since a priori no information about the relevance of dimensions is given, we initialize $\mathbf{W}$ (line 1) using non-informative weights, i.e., implicitly each entry of $\mathbf{S}$ is assumed to be $1/D$. Since in this case the subspaces for all clusters are identical, the dependency of $\mathbf{W}$ on $\mathbf{A}$ vanishes. Consequently, we can apply Def. 10.1 even without the knowledge of an initial partitioning $\mathbf{A}$. For the recomputation of the weight matrix (line 7), the subspaces per cluster might differ and we have to incorporate the actual partitioning $\mathbf{A}$. The complexity of this step is $\mathcal{O}(|E| \cdot D)$ since we have to recompute the weight for each edge, which is dominated by the computation of the norm ($\mathcal{O}(D)$).

Update of A As mentioned, the update of $\mathbf{A}$ reduces to a traditional normalized cut problem. Note that the matrix $\mathbf{W}$ and, thus, the Laplacian $\mathbf{L}$ are sparse when the underlying graph is sparse (which holds for most real graphs). Thus, we can use efficient sparse eigenvalue solvers. Additionally, multiple techniques to speed up the computation of spectral clustering for large matrices have been proposed (e.g., [YHJ09]). All these techniques can be combined with our method.

While spectral clustering has been successfully applied in many applications, it is fair to mention that this relaxation does not provide any theoretical bound on the error of the cut value [vL07]. That is, in theory, there is no guarantee that the determined cut value is close to the optimal solution. Consequently, the new partitioning might not lower the previously obtained cut value. Thus, it might be useful to add an additional termination criterion to the above algorithm as, e.g., checking whether the best cut value obtained so far has been decreased during the last m iterations or using an optimization scheme based on simulated annealing.

Update of S Even though updating $\mathbf{s}_k$ can be done for each cluster individually, the function $g_{\mathbf{a}}(\mathbf{s})$ is still hard to minimize since it is neither convex nor concave in $\mathbf{s}$. We analyzed multiple different strategies for finding local minima of this function, such as gradient descent and different greedy approaches. Here, we present only the final solution used for our approach. We selected this principle based on the following observations: a) it is very efficient to compute, b) it has obtained good normalized subspace cut values in a variety of experiments, c) the results allow an intuitive interpretation as required in many application domains. The last aspect is realized as follows: While our general model allows to use arbitrary subspace vectors, we follow the principle of most traditional subspace clustering approaches [KKZ09, AWY+99], where the relevant dimensions show uniform importance (i.e., we consider vectors as $(1/3, 0, 0, 1/3, 1/3)$). Since each dimension is either non-relevant or equally important for a cluster, an easy interpretation is possible. Formally, we consider the set

$$\mathbb{L} = \{\mathbf{s} \in [0,1]^D \mid \left(s_d > 0 \Leftrightarrow s_d = |\{i \in Dim | s_i > 0\}|^{-1}\right)\}$$

Even though the set $\mathbb{L}$ is finite, its size grows exponentially in D. Since it is intractable to enumerate and evaluate all of its members, we restrict to a meaningful subset. To ensure the selection of the most expressive dimensions, we traverse $\mathbb{L}$ starting with low-dimensional subspaces and successively expanding the best subspace with further dimensions:

1: $\mathbf{s}_{old} = 0^D$, $d = 0$ // current subspace and dimensionality
2: $\hat{\mathbb{L}} = \{\mathbf{s} \in \mathbb{L} \mid \exists_{=1} x \in Dim : s_x > 0\}$ // 1-d subspaces
3: select subspace $\mathbf{s}^* = \arg\min_{\mathbf{s} \in \hat{\mathbb{L}}} \{g_{\mathbf{a}}(\mathbf{s})\}$
4: $\mathbf{s}_{new} = (\mathbf{s}_{old} \cdot d + \mathbf{s}^*)/(d+1)$
5: if $g_{\mathbf{a}}(\mathbf{s}_{new}) < g_{\mathbf{a}}(\mathbf{s}_{best})$ then $\mathbf{s}_{best} = \mathbf{s}_{new}$
6: set $d = d + 1$ and $\hat{\mathbb{L}} = \hat{\mathbb{L}} \backslash \{\mathbf{s}^*\}$
7: goto 3 until $\hat{\mathbb{L}} = \emptyset$

This method ranks based on the set $\hat{\mathbb{L}}$, which represents the 1-dimensional subspaces. In line 4, we increase the dimensionality of the subspace vector $\mathbf{s}_{new}$ by 'adding' the next best 1-dimensional subspace s^*. The dimensionality of $\mathbf{s}_{new}$ increases in each iteration; thus, guaranteeing termination. Since the ranking in line 3 needs to be done only once, the function $g_{\mathbf{a}}$ needs to be evaluated only D times. Thus, the overall complexity for updating the subspace of a cluster is $\mathcal{O}(D \cdot t_g) = \mathcal{O}(D^2 \cdot |E|)$, where t_g is the complexity to evaluate the function $g_{\mathbf{a}}(.)$.

10.5 Experimental Analysis

Setup We compare SSCG with a variety of other partitioning clustering techniques: OptiComb [STM07], CoClus [HZZL02], SA-Clustering [ZCY09], and PICS [ATMF12] are methods using graph and feature information. We denote with SpectGraph the traditional spectral clustering using only the structural information. SpectVec1&2 is spectral clustering using only feature information. The first one uses the complete similarity graph, the second one uses the kNN similarity graph (cf. [vL07]). Proclus [AWY+99] is a subspace clustering technique for vector data. The number of clusters K and the scaling parameter θ are chosen to be identical for each method. Since SA-Clustering can only handle categorical data, for this method we discretize each numerical dimension into 10 bins. Accordingly, for PICS, which handles only binary data, numerical data is discretized into two bins. To ensure a fair evaluation, we only consider partitioning clustering approaches for our evaluation. Since overlapping clustering approaches as, e.g. [MCRE09, GFBS10] follow a completely different objective, a comparison with the methods mentioned above would always be biased to one of the paradigms. All experiments were conducted on 2.3 GHz Opteron CPUs with Java6 64-bit.

For case studies on real world data, we use graphs extracted from the DBLP database, the arXiv database, the Internet Movie Database, the German soccer

league, patent data, and gene interaction networks. We provide all datasets and their descriptions on our website. Furthermore, we generate synthetic data based on the planted partitions model [CK01]. Intuitively, given the desired number of clusters and the vertices belonging to each cluster, we randomly add edges between and within clusters according to a specified density. To generate the feature vectors, given the overall dimensionality, we randomly select a given number of relevant dimensions for each cluster. For each cluster, an individual set of dimensions is used. By default, we generate 20 dimensional data with 10 clusters, each with 100 vertices and 10 relevant features. The average density is 0.4.

For synthetic data, clustering quality is measured via the F1 value [GFM+11]. For real world data, where no ground truth is given, we use internal evaluation metrics: the normalized subspace cut (NSCut), the usual NCut considering only graph information, and the within cluster sum of squares (total distance, TD) considering only the feature information. To ensure comparability, the internal measures are always computed w.r.t. the input graph (since some approaches perform graph transformations).

Comparison on Synthetic Data We start by analyzing the effect of different kernels (Fig. 10.2). Since different kernels lead to different cut values (for the same cut), comparing the obtained cut values is unfair. Instead, we compare the clustering quality. As shown, the exponential kernel (leftmost bars) leads to the highest quality and simultaneously obtains the lowest runtime. Thus, besides being theoretically sound (Sec. 10.3.3), the exponential kernel also empirically performs best. It is therefore used for all further experiments.

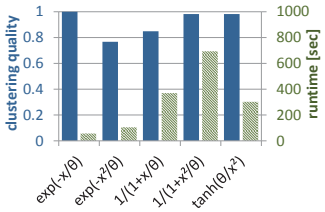


Figure 10.2: Effect of kernels

Even though our focus is on evaluating the clustering quality of SSCG, we briefly analyze the methods' efficiency. In Fig. 10.3, we increase the number of vertices in the graph. Since most methods use eigenvalue-decomposition, the slopes of their curves are in a similar range. Here, we determined the eigenvec-

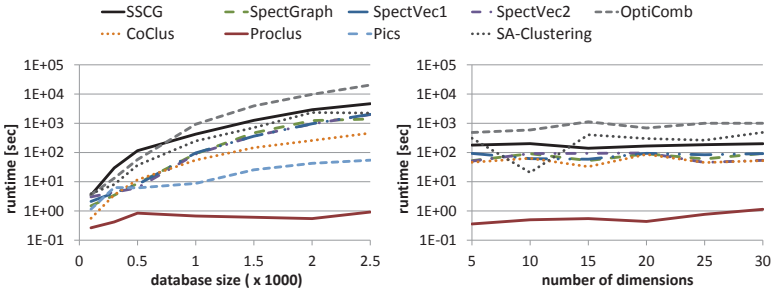


Figure 10.3: Runtime vs. database size Figure 10.4: Runtime vs. # dimensions

tors using QR-decomposition; as mentioned in Sec. 10.4, more efficient methods can be used instead. Two algorithms differ from the common shape. PICS scales slightly better than the other methods since it does not use eigenvalue decomposition. It is, however, restricted to binary data. Proclus is very efficient but, as we will see in Fig. 10.5, its clustering quality is very low.

In Fig. 10.4, we increase the dimensionality of the data. The only slight increase of the methods' runtimes indicates that the eigenvalue-decomposition (which is independent of the data's dimensionality) dominates the overall runtime. Our optimization of the subspace is very efficient.

In Fig. 10.5, we show the methods' clustering quality for an increasing database size. Only SSCG is able to detect the clustering structure. The competing approaches cannot handle data where some features are irrelevant since these features obfuscate the clustering structure in the full-space.

In Fig. 10.6, we increase the number of irrelevant dimensions per cluster. Starting with almost full-space clusters (16d), we successively lower the clusters' dimensionality (down to 2d clusters). While SSCG shows almost perfect quality, most of the other approaches decrease in their quality; the more irrelevant features, the harder to detect the clusters. SpectGraph is not affected by irrelevant features since it only uses the graph structure; though, the absolute quality is low. Interestingly, although involving feature information, OptiComb obtains almost the results of SpectGraph.

While the previous experiment has shown the effects when varying the 'quality' of the feature vectors (i.e., increasing the fraction of irrelevant features), we now analyze the methods' behavior when the structural information is distorted. In Fig. 10.7, we randomly relocate a certain number of edges. SSCG

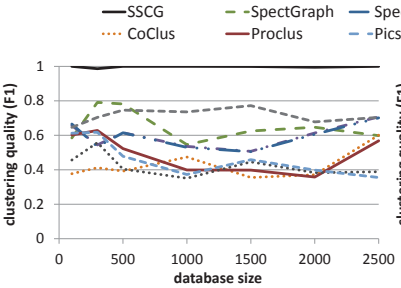


Figure 10.5: Quality vs. database size

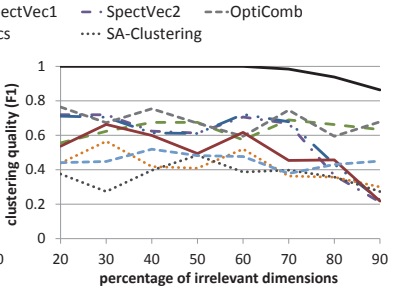


Figure 10.6: Quality vs. feature noise

is only slightly affected. By additionally exploiting the features, the results are almost stable. In contrast, the other graph based methods show a strong decrease. Obviously, the methods using only feature information are not affected (for illustration, only SpectVec2 is plotted).

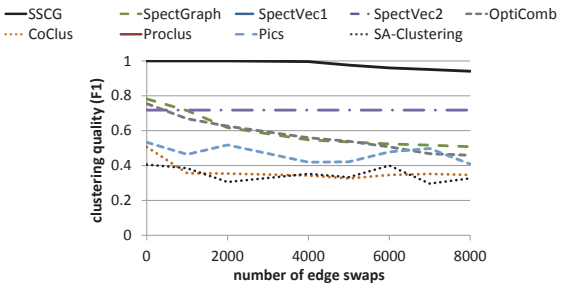


Figure 10.7: Quality vs. edge noise

Overall, SSCG's runtime is comparable to all other methods using spectral clustering and it is the only method simultaneously achieving high clustering qualities even in the presence of many irrelevant features.

Evaluation on Real World Data Since for real world data no ground truth is given, our following case studies should show two aspects: 1) The clusters found by SSCG are meaningful. We solve this issue by presenting interesting results detected by our method and by analyzing internal characteristics of the clustering result (e.g., the cut values). 2) While not being able to discuss the results of all competitors, we can examine whether a similar result than that of SSCG can already be determined by competing methods. This issue is solved by

Table 10.1: Results on DBLP

		NMI	TD	NCut	NSCut
using graph and features	**SSCG**	1.000	451.350	4.063	1.774
	CoClus	0.097	456.554	20.789	20.750
	OptiComb	0.083	461.724	21.190	22.177
	SA-Clustering	0.194	444.440	13.860	13.810
	PICS	0.103	460.138	19.234	19.943
only one data type	**SpectVec1**	0.142	432.416	17.008	10.151
	SpectVec2	0.135	436.255	17.195	9.806
	Proclus	0.104	557.686	20.984	18.465
	SpectGraph	0.395	454.823	4.355	5.848

Table 10.2: DFB data

NMI	TD	NCut	NSCut
1.000	117.448	9.313	9.313
0.200	173.977	12.661	12.445
0.483	175.953	10.507	10.548
0.395	171.361	11.515	11.500
0.223	162.488	12.236	11.966
0.184	107.954	13.291	11.581
0.209	107.482	13.038	11.098
0.202	21.3495	13.576	12.365
0.576	171.646	10.639	10.565

computing the normalized mutual information ([VEB10], NMI_{joint}) between the competitors' results to the one of SSCG. A low NMI value indicates, that SSCG is able to produce novel cluster insights, while not implying that the result of the particular competitor is bad or meaningless. An extended analysis with a pairwise comparison of all clustering results can be found on our website. To handle missing values occurring in some of the datasets due to their sparsity, the distance between features with a missing value is set to the maximal possible distance. Since this principle cannot be applied for OptiComb and PICS, missing entries are imputed here with zero values.

DBLP. In our first experiment, we analyze the DBLP data. Authors are represented by vertices and co-authorships by edges. The features consist of 20 keywords extracted from the titles of papers. The keywords are chosen to represent four different fields of research: Data-Mining, Computer Graphics, Artificial Intelligence, and Databases. They include terms like: "classification", "cluster", "graphic", and "human". We used the largest connected component (774 nodes and 1757 edges). The number of clusters is set to 24.

An interesting cluster found by SSCG is a group of 18 scientists from Max Planck Institute, TU Graz, and ETH Zurich, all established in the field of computer vision and motion capturing (left ellipse in Figure 10.8). The relevant dimensions are "motion", and "3d". Another interesting cluster is a set of 20 authors from the field of machine learning and data mining (right ellipse in Figure 10.8). They are clustered in the dimensions "cluster", "pattern", and "learning".

Table 10.1 compares all methods based on internal measures. The methods above the dashed line are the ones considering graph *and* feature information. As expected, SSCG leads to the lowest value for the NSCut. Surprisingly, even though SSCG does not minimize the (usual) NCut value, its result is better than the one of SpectGraph (which tries to optimize the NCut). Among the meth-

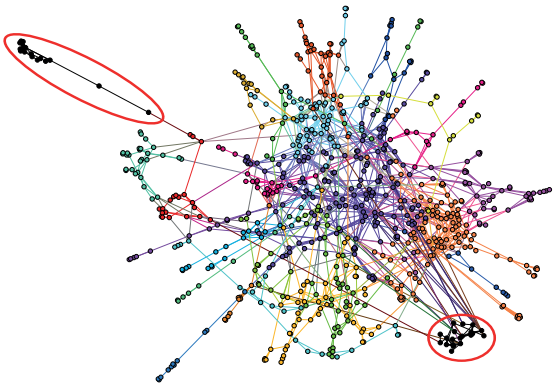


Figure 10.8: Visualization of the DBLP graph and coloring of SSCG's clusters. Two clusters are highlighted.

ods considering graph and feature information, SSCG obtains the second best TD value, while clearly outperforming these methods w.r.t. the NCut value. As indicated by the low NMI values of the competing methods, SSCG is able to reveal a novel clustering structure.

German soccer league. For our next experiment, we extract the top 100 goal getters from the German soccer league. Each node represents a player. Two players are connected if they played in the same soccer club (not necessarily at the same time). As features, we choose "number of games", "number of goals", "number of penalty kicks", "average number of goals per game", and "number of soccer clubs". The number of clusters is set to 14.

One interesting cluster is a subset of players from "Borussia Dortmund" being tightly located in the dimensions "number of clubs" (values are spread within a range of 40%) and "number of goals" (range 20%). All the other dimensions are spread across a range of at least 75%. Similarly, a subset of players from "1. FC Nuernberg" are clustered in the dimensions "number of goals" (range 8%), "number of goals per game" (range 18%), and "number of penalty kicks" (range 12%). The remaining dimensions show a spread of at least 35%. None of the competing methods was able to detect such a meaningful clustering structure.

The values of SSCG for the NSCut and the NCut are the overall best (Table 10.2). Among all approaches that use the graph structure, SSCG obtains the best feature compactness (TD). Again, the NMI value indicates that the competing methods detect different results than SSCG.

Table 10.3: arXiv1 data

		NMI	TD	NCut	NSCut
using graph and features	**SSCG**	1.000	6.027	1.617	1.420
	CoClus	0.079	5.306	16.623	16.606
	OptiComb	0.050	5.396	17.752	17.744
	SA-Clustering	0.145	9.366	11.900	11.900
	PICS	0.177	5.504	15.154	15.150
only one data type	**SpectVec1**	0.080	5.080	16.733	16.700
	SpectVec2	0.089	5.044	16.118	16.073
	Proclus	0.067	4.081	16.962	16.955
	SpectGraph	0.813	5.413	1.988	1.977

Table 10.4: arXiv2 data

NMI	TD	NCut	NSCut
1.000	18.730	1.546	0.179
0.006	15.437	8.445	8.443
dnf	dnf	dnf	dnf
dnf	dnf	dnf	dnf
0.030	22.383	6.881	6.880
dnf	dnf	dnf	dnf
0.020	15.408	7.537	7.505
0.008	17.346	8.228	8.056
0.145	15.493	2.358	2.340

arXiv. In the arXiv data, papers are represented by nodes, citations by edges, and features denote how often a specific keyword appears in the abstract of the paper. In our first experiment (arxiv1), we use the top 30 keywords and removed nodes showing no keyword, resulting in 856 nodes and 2660 edges. The number of clusters is set to 19.

SSCG found a cluster of 20 papers concerning *quantum gravity*, especially *Lorentzian and Euclidean Quantum Gravity*. The relevant dimensions of this cluster are: “space-time”, “geometry”, “gravity”, and “integral”. Another cluster consists of 16 papers concerning String Theory or more general M-Theory. The papers are about different dimensional branes, dualities and supersymmetry. The relevant dimensions are: “duality”, “point”, “dimension”, and “equation”. Here SSCG is especially meaningful since clusters contain well-known (often cited) and also highly topic relevant papers (same keywords) which makes them core papers in their fields.

As shown in Table 10.3, SSCG, again, shows the best cut values, while maintaining a reasonable distance to the competing approaches regarding TD. Considering the NMI values, most approaches find different clusters than SSCG. Only SpectGraph achieves a similar result (NMI of 0.813).

We also extract a larger citation-graph having 11,989 nodes, 119,258 edges, and 300 dimensions (arxiv2, Table 10.4). The results are in line with the observations made for the smaller dataset. While some approaches are not applicable on this data due to their extreme memory usage (larger than 7GB), SSCG can exploit the sparsity of the network which leads to a tractable eigenvalue-decomposition.

Genes. In our next experiment (Table 10.5), we analyze a gene interaction network (2,900 nodes; 8,264 edges) where genes are additionally enriched by expression values (115 dim.). While all approaches could be applied on this data, SSCG clearly outperforms them w.r.t. the NSCut value, and only SpectGraph was

Table 10.5: Gene data

		NMI	TD	NCut	NSCut
using graph and features	**SSCG**	1.000	47.939	6.702	5.539
	CoClus	0.026	50.462	15.445	15.406
	OptiComb	0.017	45.163	16.546	16.135
	SA-Clust.	0.014	46.944	16.201	16.107
	PICS	0.017	44.627	16.033	15.950
only one data type	**SpectVec1**	0.014	47.458	16.995	16.936
	SpectVec2	0.018	47.960	16.734	16.110
	Proclus	0.013	20.744	17.214	16.631
	SpectGraph	0.044	48.089	8.187	8.391

Table 10.6: IMDb data

NMI	TD	NCut	NSCut
1.000	5.963	18.650	0.563
0.053	7.812	28.516	28.516
0.125	7.776	28.169	27.924
0.182	9.351	22.169	22.169
0.149	7.768	26.257	26.255
0.161	7.648	27.813	27.374
0.168	7.624	27.684	27.217
0.166	5.278	27.839	27.486
0.282	7.816	16.349	16.256

able to realize a similar NCut score. For this data, we observe the largest differences in the clustering results. A pairwise analysis of all clusterings reveals that none of them can be considered similar, indicating that this dataset is particular challenging to cluster.

Internet Movie Database. The next dataset is an extract of the IMDb. We use movies with at least 200 rankings and an average ranking of at least 6.5 as nodes. Two movies are connected if they share actors or if there exists a reference (e.g., spoofs or follow ups) to each other. As features, we choose all 21 movie genres. To allow good interpretation, we focus on movies produced in USA, Canada, UK, or Germany. We use the largest connected component (862 nodes and 4388 edges) and look for 30 clusters.

SSCG found a cluster of 19 movies including “Evil Dead II”, “Poltergeist”, and “Predator” based on the relevant dimensions: “Horror”, “Mystery”, and “Thriller”. Another interesting cluster contains 19 movies concerning Jimi Hendrix, Chuck Berry, and U2. All movies are either biographies or movies about music. Conveniently, the relevant dimensions are: “Biography” and “Music”. Finally, there is a cluster of 20 romantic comedies containing movies like “10 Items or Less”, “Driving Miss Daisy”, and “Feast of Love”. These three movies are connected through the actor Morgan Freeman. “Comedy”, “Romance”, and “Drama” are the relevant dimensions. As shown in Table 10.6, SSCG obtains the overall best NSCut value. Considering all methods that are using the network structure and the feature information, SSCG also achieves the best values for TD and NCut.

Patents. Finally, we want to show the applicability of SSCG on a large citation network of patents with 100,000 nodes, 188,631 edges, and 5 dimensions. Most of the methods were not applicable on this data. Particularly, from the methods considering graph and feature information, only SSCG and PICS could be applied. The clustering of SSCG showed the following properties (TD: 22534, NCut: 1.04,

NSCut: 0.70), which clearly outperforms the result of PICS (TD: 24411, NCut: 10.18, NSCut: 9.94) w.r.t. all measures. This difference of the clustering results is also indicated by a low NMI value of 0.060.

Overall, as shown by all experiments, SSCG is able to detect meaningful clusters on a variety of real world datasets. The competing approaches generate highly different results. The internal evaluation measures show that the clusters of SSCG are very compact in the feature space (low TD) and also well separated in the graph (low cut values).

10.6 Conclusion

In the proposed spectral clustering method for graphs with feature vectors, we integrated the subspace clustering principle, where we tackle the problem of irrelevant features that possibly differ for each cluster. As a consequence, the affinity matrix is not given a priori but depends on the partition as well as the determined features and is, thus, part of the learning process. To tackle the fundamental challenge of comparing the clustering structures for different feature subsets, we defined an objective function that is unbiased w.r.t. the number of relevant features. For efficiently approximating our objective, we developed the algorithm SSCG based on spectral clustering. SSCG was applicable on large datasets and was the only method achieving high clustering qualities in the presence of many irrelevant features. For a variety of real-world datasets, SSCG was able to detect meaningful clusters which are very compact in the feature space and, simultaneously, well separated in the graph.

11

Modularity for Subspace Clustering in Multi-Dimensional Graphs

In contrast to the previous chapter, where we considered graphs with multi-dimensional vertex labels, in this chapter, we will focus on graphs with multi-dimensional edge weights, i.e., the same vertices can share different types of relations. While traditional clustering approaches for graphs with multi-dimensional edge information consider all dimensions as equally important, recent advances indicate a varying relevance of dimensions for different clusters. Especially in the presence of many different relations, it is crucial to be able to detect clusters of vertices that are densely connected in only a subset of the relation types.

Modularity is one of the most sensitive and best known quality functions to express the strength of communities. In this work, we extend this widely used optimization criterion for multi-dimensional edge weights by following the principles of subspace clustering. Our modularity extension can already be adopted by some of the existing optimization approaches. To deal more effectively with the extended search space due to the variance of the dimensions' relevance, we propose the efficient clustering algorithm SuMo for clustering networks based on the subspace modularity.

11.1 Introduction

For real-world applications, besides the mere relational information, additional information for the graph data is available and is useful to consider for clustering the vertices. In the previous chapter, we proposed a new partitioning approach for graphs with additional vertex information. In this chapter, we will instead consider graphs with additional information on the *edges*. It is important to note, that for most cases it is not possible to transfer an edge labeled graph into a vertex labeled one (Section 6.3 in [Bod14]), such that we cannot simply use the technique presented in Chapter 10 to solve the problem at hand. For graphs with annotated edge information, we can differentiate between two types of edge labels. They can either represent characteristics, e.g., common interests for two individuals in a social network, or edge labels can represent edge weights denoting the strength of the relation between two vertices. In a social network, the ties between two individuals can be of different strength, similarly, the collaboration between two scientists in a co-authorship network might be weighted by the number of co-authored papers. In the following, we will focus our considerations to edge weights.

Acknowledging that the ignorance of available edge weights abandons a lot of potentially useful information, various approaches have already been adjusted to handle edge weights [New04a]. For similar reasons of maximally utilizing the offered information, the community recently launched mining problems for networks with multiple types of edges or simply multi-dimensional edge weights. In complex systems, the entities' relations are usually multifaceted, with different aspects potentially leading to different weightings. In a social network, relations between the individuals can be weighted according to their proximity, the number of mutual interests, the intensity of communication, or the number of shared friends. In a co-author network, the co-authorship can be differentiated for different research areas, the relation can also be weighted based on mutual citations, or whether the two individuals have been co-workers. In the simple co-author network depicted in Fig. 11.1, each dimension corresponds to a certain keyword and weights represent the number of co-authored papers of two authors containing this keyword.

Often the number of potential edge weight dimensions is large and while using multiple information sources is meaningful in general, it also entails the risk that some of the weight dimensions might not support or even disagree with the

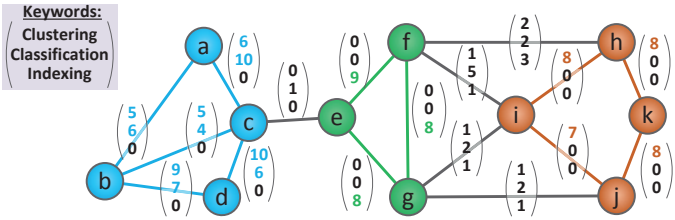


Figure 11.1: Example co-authorship network

underlying clustering structure. For example, for a co-author network such as in Fig. 11.1, we do not expect to find clusters of authors that have common papers containing all the keywords, especially if we have a large set of keywords. Instead, each author cluster will only have a certain set of relevant keywords, characterizing the group's main research field. The problem of a high proportion of irrelevant, misleading information is well documented in the context of the curse of dimensionality [BGRS99], especially for the task of clustering [HKK+10]. To avoid the obfuscation of the clustering through irrelevant dimensions for traditional vector data, the paradigm of subspace clustering identifies clusters only in the context of their relevant features. In our scenario with multi-dimensional edge weights, the goal is to group vertices such that for each group a subset of relevant dimensions exists for which the sum of internal edge weights is high and edges between clusters are sparse and only lowly weighted.

In this work, we extend one of the most widely used quality functions for graph clustering ([For10]) to handle multi-dimensional edge weights with locally irrelevant dimensions. The most popular and established techniques for graph clustering are spectral clustering methods and techniques optimizing the modularity. Since methods for detecting community structures usually assume that the network naturally divides into subgroups, the number and size of clusters is inherently determined by the network and is not known a priori. This characterizes the superiority of modularity-based techniques over spectral clustering, where the number of clusters has to be known in advance. We, therefore, present an adaptation of the modularity optimization criterion to handle locally irrelevant dimensions. We show how certain existing algorithms can already be used to approximate the optimal clustering w.r.t. our adapted modularity. In addition, we present our algorithm SuMo which exploits a more informed search strategy based on the eigendecomposition of the modularity gain matrix. We evaluate our solution on synthetic and real-world data.

11.2 Related Work

For clustering graph data, various models and techniques have been developed. An overview of these techniques is given by Fortunato [For10]. Many graph clustering techniques were developed for simple graph data (without multiple dimensions). Established approaches include algorithms minimizing cuts (e.g., the Kernighan-Lin algorithm [KL70]), spectral clustering algorithms (e.g., [ST96]), and modularity-based approaches. Many of those approaches consider weighted graphs. While the aforementioned approaches partition the graph, there also exist approaches detecting overlapping dense subgraphs.

The modularity measure was first introduced for unweighted graphs by Newman and Girvan [NG04]. They also proposed a first top-down algorithm based on *edge-centrality*. In [New04a], the modularity measure was generalized to also consider weighted networks. A multitude of algorithms for modularity optimization have been proposed since, e.g., [New04b, CNM04, New06]. An efficient two-step algorithm detecting good modularity maxima was proposed by Blondel et al. [BGLL08]. Based on a first clustering determined by the first step, the next step creates a new graph by replacing the clusters with vertices. The two steps are iterated as long as the clustering changes, which results in a clustering hierarchy where each level represents a certain resolution level.

Recently, several approaches for clustering graphs with several dimensions were proposed. In some cases, such networks are addressed as multi-dimensional [TWL12] networks. The approach of [CZ09] clusters each graph separately and combines the clusterings in a post-processing step using an ensemble approach. The basic idea of other approaches [DFVN12, TLD09, KRDI10, BCG11] is to combine the information from the different graphs and to apply existing clustering methods (for one-dimensional graphs) to the combined representation. A modularity-based spectral algorithm was proposed by Tang et al. [TWL12]. Although some approaches enable a global weighting of each dimension, none of these approaches considers clusters in subspaces of the dimensions. As the approaches combine the information of the different graphs before the actual clustering, it is not straight-forward to extend them to detect subspace clusters. In [BGHS12, BGHS13], approaches for clustering graphs with multi-dimensional edge attributes are proposed, which also consider subspaces. However, in this case the edge attributes are not weights, but *feature vectors* of the relation. Thus, the task is to find clusters with *similar* attribute values. This solution cannot be

transferred to our problem of finding clusters with high edge weights. Similarly, in the method by Qi et al. [QAH12], labels represent "edge content" (feature vectors extracted from text). In this approach, the *edges* of the graph are clustered by a partitioning approach based on connectedness and similarity of the edge content, without considering subspaces. From these edge clusters, overlapping communities of nodes are obtained. So far, there exists no approach tackling the challenge of detecting clusters with varying locally irrelevant dimensions in graphs with multi-dimensional edge weights.

11.3 Subspace Modularity

Modularity is the most popular quality function for evaluating the strength of communities and partitions in graphs. In this section, we introduce a straightforward extension of the modularity measure to include the evaluation of subspace projections. For the following discussions, we consider undirected graphs without multi-edges, whose edges are associated with multi-dimensional weight vectors.

Definition 11.1 (Graph) *A graph G is defined as a triple $G = (V, E, f_E)$ with a set of n vertices $V = \{v_1, \ldots, v_n\}$, a set of edges $E \subseteq M := V \times V$, a set of dimensions D and a function $f_E : M \to \mathbb{R}_{\geq 0}^{|D|}$, $|D| \in \mathbb{N}_{>0}$ such that:*

$$f_E(E) \subseteq \mathbb{R}_{\geq 0}^{|D|} \quad \wedge \quad \forall e \in M \setminus E : f_E(e) = \mathbf{0}_{|D|}$$

The function f_E assigns a $|D|$-dimensional vector $\mathbf{w}_{i,j} := f_E(v_i, v_j)$ with non-negative components to each adjacent vertex pair v_i, v_j and the zero-vector to each non-adjacent vertex pair. For this *edge weight vector* the d-th component is the *edge weight in the d-th dimension* ($d \in D$) and denoted by $w_{i,j}^d$. W.l.o.g., we assume that there is at least one non-zero edge weight in each dimension and therefore at least one edge in the graph. (Dimensions without edges have no clustering structure and can be excluded from further consideration.)

A clustering for a graph G is commonly defined as a partitioning of the graphs' vertices. Since we consider clusters in subspace projections, we additionally assign an individual set of dimensions to each cluster:

Definition 11.2 (Clustering) *For a graph $G = (V, E, f_E)$, $\forall K \in \mathbb{N}, 1 \leq K \leq |V| :$ $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} := \Big(C_1, \ldots, C_K \mid C_1, \ldots, C_K \subseteq V,\ \bigcup_{k=1}^{K} C_k = V,\ \bigcap_{k=1}^{K} C_k = \{\}\Big)$, $\mathcal{S} := (S_1, \ldots, S_K \mid S_1, \ldots, S_K \subseteq D \wedge S_1, \ldots, S_K \neq \{\})$ is a* clustering *of G and each $C_k \in \mathcal{C}$ is called a* cluster.

Since obviously not every partitioning is a good clustering, we need an objective function to assess the quality of a graph clustering. The most widely used one is the modularity [For10], which we briefly discuss before extending it to consider subspaces.

11.3.1 The Existing Modularity Measure.

While there exist multiple definitions of what constitutes a good clustering within a graph, the unifying idea is that a good clustering should have a relatively high density of edges inside each cluster and a low edge-density between different clusters. The modularity helps to quantify *low* and *high* density by measuring the degree to which the arrangement of edges identified by the clustering is statistically surprising compared to a null model, i.e., an equivalent graph (same size and vertex degrees) where edges are placed at random. For unweighted graphs, the modularity corresponds, up to a normalization factor, to the number of intra-cluster edges minus the expected number of intra-cluster edges in the null model. The weighted version of the modularity measure for a clustering $\mathcal{C}$ is defined as $Q(\mathcal{C}) := \frac{1}{w} \sum_{C_k \in \mathcal{C}} \sum_{v_i, v_j \in C_k} \left[w_{i,j} - \frac{w_i \cdot w_j}{w} \right]$ where $w_i := \sum_{j=1}^{n} w_{i,j}$ and $w := \sum_{i=1}^{n} w_i$. This generalization of the modularity for weighted edges favors clusterings with a high density of high-weighted edges inside each cluster and a low density of such edges between clusters. Modularity values lie in $(-1, 1)$, where positive values indicate a possible presence of a clustering structure and the larger the values, the more significant is the clustering.

11.3.2 The Subspace Modularity Measure.

In the following, we aim to further extend the modularity to multi-dimensional edge weights and, more importantly, to enable the evaluation of clusters in the context of only the relevant edge weight dimensions. Especially when the weight-vector is high-dimensional, the probability of all weight-dimensions being relevant for each cluster decreases. A high proportion of irrelevant or noisy dimensions can obfuscate the clustering structure and an otherwise meaningful clustering result will achieve misleadingly low quality values when evaluating the modularity with respect to all dimensions. To avoid this misinterpretation of clusterings due to dominating irrelevant dimensions, it is crucial to restrict the evaluation only to relevant dimensions.

A further important aspect is that for each cluster a different set of dimensions can be relevant. For example, in our example from Fig. 11.1, each group of authors works on a different topic and thus different keywords are relevant for each cluster. Intuitively, a dimension should be relevant for a cluster if and only if its edge weights in this dimension are larger than expected. In Fig. 11.1, we intuitively have three different clusters: cluster $C_1 = \{a, b, c, d\}$ with the relevant dimensions 1 and 2, $C_2 = \{e, f, g\}$ in dimension 3, and $C_3 = \{h, i, j, k\}$ in dimension 1. However, if we simply summed up the weights of each edge to a single edge weight in order to use the existing modularity measure, the clusters C_2 and C_3 would be merged into a single cluster, even though this cluster would not be well connected in any of the single dimensions. In contrast, with our subspace modularity measure, we are able to detect all three clusters and their corresponding subspaces.

To incorporate multi-dimensional edge weights into the modularity, we intuitively simply sum up the modularity contributions of every edge-dimension in the subspace of the corresponding cluster. To ensure comparability of the weights in different dimensions, we assume the weights in each dimension to be normalized to $[0, 1]$.

Definition 11.3 (Subspace Modularity) *Let G be a graph and $\mathcal{C}$ a clustering of G. The subspace modularity Q_D is defined as follows:*

$$Q_D(\mathcal{C}, \mathcal{S}, G) = \frac{1}{w} \sum_{C_k \in \mathcal{C}} \sum_{d \in S_k} \sum_{v_i, v_j \in C_k} \Bigg[w_{i,j}^d - \underbrace{\frac{w_i \cdot w_j}{|D| w}}_{=:\mu_{i,j}^d} \Bigg]$$

where $w_i := \sum_{d=1}^{|D|} \sum_{j=1}^{n} w_{i,j}^d$ *and* $w := \sum_{i=1}^{n} w_i$.

For each Cluster C_k, we compare all edge weights $w_{i,j}^d$ in each relevant dimension $d \in S_k$ against their expected weight $\mu_{i,j}^d$. At this point, one could argue that the expected weight should be determined based on each dimension individually, i.e., $\mu_{i,j}^d = w_i^d \cdot w_j^d / \sum_{i=1}^{n} w_i^d$. Although this would be a formally correct extension of the modularity, given the subspace clustering perspective, we run into the following problem. Since each cluster can have its individual set of relevant dimensions, some dimensions might be important for more clusters than others. This results in different weight distributions for each dimension and prevents a comparability of the modularity contributions in different dimensions. If, e.g., for one edge we have $w_{i,j}^{d1} = w_{i,j}^{d2}$, $w_i^{d1} = w_i^{d2}$, and $w_j^{d1} = w_j^{d2}$, then, in the world of subspace clus-

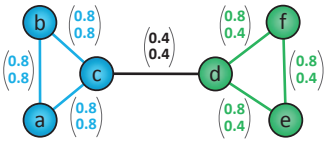


Figure 11.2: Graph with two clusters

tering, we expect that both dimensions, $d1$ and $d2$, are equally important for this cluster, independent of their weights in other clusters. For example, the graph in Fig. 11.2 consists of the two clusters $C_1 = \{a, b, c\}$ and $C_2 = \{d, e, f\}$, both having the relevant dimensions $\{1, 2\}$. As the edges in C_1 have equal values in both dimensions, we argue that they also should have the same influence in both dimensions. However, if we determined $\mu^d_{i,j}$ based on each dimension individually, the influence in $d1$ would be lower than in $d2$, as the sum of all edge weights in $d1$ is higher. Therefore, in our definition the expected edge weight is the same for every dimension, and depends on the sum of the edge weights for all dimensions. It is important to remember, that the edges' weights in each dimension are normalized to $[0, 1]$. The expected edge weight is normalized by the number of dimensions and the overall weight w. Through the overall normalization factor $\frac{1}{w}$, we guarantee that the subspace modularity only reaches values in $(-1, 1)$. This allows a comparison among clusterings for different graphs or with different cluster counts.

With this adaption of the modularity measure, it is possible to evaluate a set of partitions and their associated subspaces for a graph with multi-dimensional edge weights. The question at hand is how to determine the optimal clustering $(\mathcal{C}^*, \mathcal{S}^*)$, i.e., the clustering for which the subspace modularity $Q_D(\mathcal{C}^*, \mathcal{S}^*, G)$ is maximal. We call it the Maximum Subspace Modularity (**MSM**) problem.

Definition 11.4 (Maximum Subspace Modularity) *Given a graph $G = (V, E, f_E)$ the maximum subspace modularity (MSM) is the problem of finding a clustering $(\mathcal{C}^*, \mathcal{S}^*)$ out such that*

$$(\mathcal{C}^*, \mathcal{S}^*) = \underset{(\mathcal{C},\mathcal{S})\in\mathfrak{C}(G)}{\operatorname{argmax}} \{Q_D(\mathcal{C}, \mathcal{S})\}$$

where $\mathfrak{C}(G)$ is the set of all possible clusterings of G according to Definition 11.2.

Since already the decision problem version of the traditional unweighted modularity is known to be NP-complete [BDG+06], we cannot expect a generalization of this problem to be optimized efficiently.

11.3.3 Subspace Modularity Complexity Analysis.

For analyzing the complexity of optimizing the subspace modularity measure of Definition 11.3, we first formulate the according decision problem version of our optimization problem MSM. By showing that the decision problem version of the classical modularity for weighted graphs can be reduced to our decision problem in polynomial time, we can show that our problem is NP-hard. The decision problem version to the MSM optimization problem is formalized by:

Definition 11.5 (Decision problem for MSM) *Given a graph $G = (V, E, f_E)$ and a constant c, is there a clustering $(\mathcal{C}, \mathcal{S})$ with $Q_D(\mathcal{C}, \mathcal{S}) \geq c$?*

Theorem 11.1 *The decision problem version of MSM is NP-complete.*

Proof 11.1 *a) The input for the maximal modularity problem for weighted graphs (MWM) is a graph $G = (V, E, f_E)$ with one-dimensional edge weights $f_E(E) \in \mathbb{R}$. Since we only have 1-dimensional weights and each subspace must have a cardinality of ≥ 1, the subspace modularity of Definition 11.3 corresponds to the traditional weighted modularity. Thus, since MWM is NP-complete [BDG+06], MSM is NP-hard.*
b) MSM is in NP: For a given clustering $(\mathcal{C}, \mathcal{S})$, we can check in polynomial time $O(|V|^3 \cdot |D|)$ whether $Q_D(\mathcal{C}, \mathcal{S}) \geq c$. Since the verification is in P, MSM is in NP.
c) combining a) and b) $\Rightarrow$ MSM is NP-complete.

11.4 Algorithm

Although our MSM problem has the same complexity class as the original weighted modularity problem, it seems to be more complex since the search space is enlarged exponentially by considering possible subspaces. A closer look, however, reveals, that, in order to maximize the subspace modularity, a subspace S_k for a cluster C_k should contain all and only those dimensions whose modularity contributions are positive. For a given clustering $\mathcal{C}$, the optimal subspaces can directly be derived for each cluster $C_k \in \mathcal{C}$ by:

$$S(C_k) = \{d \in D \mid \sum_{v_i, v_j \in C_k} (w_{i,j}^d - \frac{w_i \cdot w_j}{|D|w}) > 0\} \tag{11.1}$$

Accordingly, we will denote by $\mathcal{S}(\mathcal{C})$ the tuple of subspaces containing for each cluster $C_i \in \mathcal{C}$ a subspace $S_i := S(C_i) \in \mathcal{S}(\mathcal{C})$. Since the subspaces can now

directly be inferred from a given partitioning of the vertices, the search space reduces to that of the original modularity problem, namely finding the optimal partitioning. Numerous optimization approaches for the modularity have been proposed over the years. Spectral methods (e.g., [New06]) work on a "modularity" matrix based on the adjacency matrix of the graph. These approaches can not easily be adapted for our problem as a single adjacency matrix does not well represent a multi-dimensional graph. An aggregation of all matrices into a single one would hinder the detection of clusters in subspaces. However, several greedy approaches for modularity can be adapted to our problem. Often the idea is to generate a hierarchical sequence of clusterings, either agglomerative or divisive, and to stop if there is no further modularity gain or to choose the clustering with maximal modularity score out of the complete sequence in the end. In each iteration, the decision for merging or splitting clusters is based on the modularity gain of the different alternatives. Other approaches (e.g., [BGLL08]) iteratively move single vertices to other clusters such that the modularity is increased. Such local reassignments are also part of approaches based on simulated annealing or extremal optimization.

While these approaches prove to be very effective in the case of just a single weight per edge, the subspace determination for multi-dimensional edge weights complicates their direct adaption for our problem. The difficulty here is that for the initially very small clusters (in case of an agglomerative approach), or the initially very large clusters (in case of an divisive approach), or simply just the bad clusters of initial random partitionings, the modularity contribution of all dimensions is usually negative. According to our straight-forward subspace determination in Equation 11.1 all subspaces would be empty, resulting in a subspace modularity score of zero. As a consequence, there is no basis for decision-making to merge or split clusters in the first iterations, which are then performed completely at random. Since in this preliminary state of the hierarchical clustering there is no meaningful basis for choosing a set of relevant dimensions, the decision for the hierarchical sequence initially has to be guided by all dimensions. With continuing iterations, the relevant dimensions become more apparent, and should have a stronger influence on the clustering sequence. For utilizing hierarchical clustering approaches to optimize the subspace modularity, we need an objective function initially incorporating all dimensions, increasing the influence of the relevant dimensions with progressing iterations, and thus converging to the actual subspace modularity function:

Definition 11.6 (α-Modularity) *Let G be a graph, $\mathcal{C}$ be a clustering of G, $t \in \mathbb{N}_{\geq 0}$ the number of iterations, and $\alpha : \mathbb{N}_{\geq 0} \to [0,1]$. The α-modularity Q_D^α is defined as the following clustering objective function:*

$$Q_D^\alpha(\mathcal{C}, G, t) = \frac{1}{w} \sum_{C \in \mathcal{C}} \sum_{v_i, v_j \in C} \sum_{d \in S(C)} \alpha(t) \cdot \left(w_{i,j}^d - \frac{w_i \cdot w_j}{|D| w} \right) + \frac{1}{w} \sum_{C \in \mathcal{C}} \sum_{v_i, v_j \in C} \sum_{d \notin S(C)} (1 - \alpha(t)) \cdot \left(w_{i,j}^d - \frac{w_i \cdot w_j}{|D| w} \right)$$

Basically, this α-Modularity introduces a weighting of relevant and irrelevant dimensions. Since the initial influence of the irrelevant dimensions should fastly be diminished such that the objective function fastly converges to the original subspace modularity, the exponential function is perfectly suited for weighting:

$$\alpha(t) := 1 - 0.5 \cdot e^{-p \cdot t \cdot \frac{1}{|V|}} \quad , p \in \mathbb{R}_{\geq 0}$$

This α-function ensures that we start with $\alpha(t = 0) = 0.5$ in the first iteration and then converge relatively fast to 1 ($\alpha(t \to \infty) = 1$). Since the length of the hierarchical clustering sequence and, thus, the number of iterations after which we yield meaningful subspaces strongly depends on the size of the underlying graph, $\alpha(t)$ also depends on the number of vertices $|V|$. The constant p influences how fast $\alpha(t)$ converges against 1 and is called the *convergence speed factor* of α. In experimental results the value of $p = 1.5$ has shown good results and is used as default value for our evaluation.

With the help of the α-Modularity of Definition 11.6, we can simply apply the greedy clustering procedures based on iterative reassignments of vertices that are already available in the literature. In the following, we exemplarily describe how we can adapt the approach from [BGLL08] and a simple hierarchical bottom-up approach. Algorithm 11.1 describes the general greedy workflow adapted from [BGLL08]. We iteratively perform a hierarchical clustering step based on the α-Modularity (line 3), until the clustering quality measured as subspace modularity does not improve any more. In this case, we collapse the graph (line 7), according to the second phase described in [BGLL08], which constructs a new graph, where nodes represent the clusters of $\mathcal{C}^t$ and edges and weights are adapted from G^t. Based on this new graph, we continue the iterative clustering update until neither the clustering nor the graph changes anymore. The function for the $clustering\text{-}update(\mathcal{C}^t, G^t, Q_D^\alpha, t)$ can be any greedy iterative clustering method guided by the α-Modularity objective function. In [BGLL08], this step is designed as a greedy

Algorithm 11.1: Framework for hierarchical clustering

input : A graph G
output: A partitioning $\mathcal{C}$ of the graph G

```
1 t = 0; G^t = G; C^t = singleton clusters of V
2 while true do
3     C^{t+1} = clustering-update(C^t, G^t, Q_D^α, t)
4     if Q_D(C^{t+1}, S(C^{t+1}), G) ≥ Q_D(C^t, S(C^t), G) ∧ C^{t+1} ≠ C^t then G^{t+1} = G^t
5     else
6         C^{t+1} = C^t
7         G^{t+1} = collapse(G^t, C^t)                    /* see [BGLL08] */
8         if G^t == G^{t+1} then break
9     t = t + 1
```

Algorithm 11.2: $clustering\text{-}update(\mathcal{C}^t, G^t, Q_D^\alpha, t)$ as local reassignment of vertices according to [BGLL08]

input : A clustering $\mathcal{C}$, a graph G, an objective function Q, an iteration counter t
output: An updated partitioning $\mathcal{C}^*$ of G

```
1 C* = C
2 forall the v ∈ V do
3     Find cluster C incident to v with highest quality gain if v is
      transferred to C
4     if Q(C*_{C_k ← v_i}, G, t) ≥ Q(C*, G, t) then C* = C*_{C_k ← v_i}
```

local approach (Algorithm 11.2) where one vertex v is reassigned to a neighboring cluster C if this leads to a positive α-Modularity gain out of all choices for C.

A corresponding adaption of a traditional hierarchical bottom-up approach is depicted in Algorithm 11.3. Here, we iteratively combine the cluster pair $(C_i, C_j) \in \mathcal{C} \times \mathcal{C}$, whose union yields the highest positive α-Modularity gain.

11.4.1 The SuMo Algorithm

Although the presented simple greedy heuristics are already able to find a good clustering solution with respect to the subspace modularity, we want to introduce a new $clustering - update$ function SuMo, incorporating in each step all of the available information. We will see in the experimental section that SuMo is more robust against noise dimensions than Algorithms 11.2 and 11.3, which is a neces-

Algorithm 11.3: *clustering-update*$(\mathcal{C}^t, G^t, Q_D^\alpha, t)$ as pairwise merge of clusters

input : A clustering $\mathcal{C}$, a graph G, an objective function Q, an iteration counter t
output: An updated partitioning $\mathcal{C}^*$ of G

```
C* = C
while true do
    choose arg max_{(C_i,C_j)∈C*×C*} Q(C*_{C_i←C_j}, G, t)
    if Q(C*_{C_i←C_j}, G, t) ≥ Q(C*, G, t) then C* = C*_{C_i←C_j}
    else break
```

sary property for the task of subspace clustering. While Algorithm 11.2 just tries to move any vertex v to an incident cluster, SuMo tries to find the "most promising" vertex-cluster pair in each step. Instead of just deciding based on the quality gain of moving one vertex v into a cluster C, the key idea for SuMo is to additionally consider the tendency of neighboring vertices to follow into cluster C in later iterations. Based on the definition of the modularity, the contribution of a vertex to the modularity of a clustering strongly depends on the cluster membership of its incident vertices. Thus, moving a vertex v_i whose majority of neighbors will later join its cluster choice might be preferable compared to a node v_j whose neighbors will keep their current cluster assignments, even if the actual quality gain by just moving v_j is larger than that of moving v_i. Of course, this preference is also influenced by the quality gain expected from the neighbors of v_i.

First, we consider for each cluster C_k and each vertex v_i the quality gain of moving v_i into C_k and for each neighbor v_j of v_i the additional quality of moving v_j as well. We represent this as a *quality gain matrix*:

Definition 11.7 (Quality Gain Matrix of a Cluster) *Given a graph G, a clustering $\mathcal{C}$ of G, a cluster $C_k \in \mathcal{C}$, and a quality function Q. The quality gain matrix $\boldsymbol{G}_k$ is then defined as:*

$$\boldsymbol{G}_{k,i,j} = \begin{cases} Q(\mathcal{C}_{C_k \leftarrow \{v_i\}}) - Q(\mathcal{C}) & \text{if } i = j \\ \frac{1}{md} \cdot \left[Q(\mathcal{C}_{C_k \leftarrow \{v_i, v_j\}}) - Q(\mathcal{C}_{C_k \leftarrow \{v_i\}}) \right] & \text{if } v_j \in N(v_i) \\ 0 & \text{else} \end{cases}$$

where $N(v_i) = \{v_j \in V | \exists \{v_i, v_j\} \in E\}$ is the neighborhood of v_i, $\mathcal{C}_{C_k \leftarrow M}$ represents the clustering $\mathcal{C}$, where all vertices $M \subseteq V$ are moved to cluster C_k, and $md = \max_{u \in \cup_{v_i \in C_k} N(v_i) \setminus C_k} deg(u)$ is the maximal degree of neighboring vertices of C_k.

A row $\mathbf{G}_{k,i}$ of the quality gain matrix represents the quality gains of moving v_i in combination with its neighbors $N(v_i)$. Since we are actually moving just the vertex v_i, the quality gains of its neighbors are only of secondary importance, which requires the weighting by $\frac{1}{md}$. Otherwise, the beneficial impression of a vertex could mainly originate from its neighbors. At this point one could simply choose the vertex v_i and the cluster C_k for which $C_k, v_i = arg\,max_{C_k \in \mathcal{C}, v_i \in V} \left(\sum_{v_j \in V} \mathbf{G}_{k,i,j} \right)$. This would be a valid choice given two conditions: a) all of v_i's neighbors actually follow into cluster C_k and b) it is beneficial for cluster C_k to absorb all of v_i's neighbors. Since, intuitively, these two conditions do not necessarily hold, our objective function needs to incorporate two different perspectives: the preferences of each vertex to join the different clusters and the preferences of each cluster to absorb the different vertices.

For the perspective of the vertices, the preference of a vertex v_i for a cluster $C_k \in \mathcal{C}$ can simply be represented by the relative quality gain $\frac{\mathbf{G}_{k,i,i}}{\sum_{C_l \in \mathcal{C}} \mathbf{G}_{l,i,i}}$. To capture the desirability for cluster C_k to include a vertex v_i, we define a preference vector $\mathbf{t}_k$. Intuitively the preference for a vertex v_i depends on the quality improvement gained by including v_i ($\mathbf{G}_{k,i,i}$), on potential further improvements achieved by including its neighbors v_j ($\mathbf{G}_{k,i,j}$), and on the cluster's preference of including these neighbors ($\mathbf{t}_{k,j}$). Expressed formally, we have the following eigenvector problem:

$$\lambda \cdot \mathbf{t}_k = \mathbf{G}_k \cdot \mathbf{t}_k$$

For a better comparability of the preference values for different vertices, we want to ensure that each entry of $\mathbf{t}_k$ has the same sign. By slightly adapting the quality gain matrix $\mathbf{G}_k$ to a positive matrix, we can ensure that there exists a positive eigenvector corresponding to the largest eigenvalue (Perron-Frobenius theorem [Mey00]). Therefore, we adapt $\mathbf{G}_k$ as follows:

$$\mathbf{G}^*_{k,i,j} := \begin{cases} \mathbf{G}_{k,i,j} + |m| & \text{if } v_j \in N(v_i) \cup \{v_i\} \\ \epsilon & else \end{cases}$$

where $m = \min_{k,i,j} \{\mathbf{G}_{k,i,j}, 0\}$ and $\epsilon \in \mathbb{R}^+, \epsilon \ll 1$ is an arbitrarily small positive number. The eigenvector $\mathbf{t}^*_k$ corresponding to the largest eigenvalue of $\mathbf{G}^*_{k,i,j}$ represents the vertices' desirability for one cluster C_k. To enable a comparability between the preference vectors of different clusters, we normalize each vector $\mathbf{t}^*_k$ such that its largest entry is 1. We combine the clusters' and the objects' preferences into one "probability" vector $\mathbf{p}_k \in [0,1]^{|V|}$ for each cluster:

Algorithm 11.4: $clustering\text{-}update(\mathcal{C}^t, G^t, Q^{\alpha}_D, t)$ as local reassignment of vertices according to SuMo

input : A clustering $\mathcal{C}$, a graph G, an objective function Q, an iteration counter t
output: An updated partitioning $\mathcal{C}^*$ of G

1 $\mathcal{C}^* = \mathcal{C}$
2 **forall the** $C_k \in \mathcal{C}$ **do** Compute $\mathbf{t}_k$, $\mathbf{p}_k$, $\mathbf{j}_k$
3 **while** $P = \{(v_i, C_k) \mid v_i$ *not moved yet*$, v_i \notin C_k\} \neq \emptyset$ **do**
4 Choose pair $(v_i, C_k) \in P$ for which $\mathbf{j}_{k,i}$ is maximal
5 **if** $Q(\mathcal{C}^*_{C_k \leftarrow v_i}, G, t) \geq Q(\mathcal{C}^*, G, t)$ **then**
6 $\mathcal{C}^* = \mathcal{C}^*_{C_k \leftarrow v_i}$
7 $\mathbf{p}_{k,i} = 1$ and $\mathbf{p}_{l,i} = 0 \; \forall l \neq k$, recompute $\mathbf{j}_k$'s
8 **else break**

$$\mathbf{p}_k = \left(\frac{\mathbf{G}^*_{k,i,i} \cdot \mathbf{t}^*_{k,i}}{\sum_{C_l \in \mathcal{C}} \mathbf{G}^*_{l,i,i} \cdot \mathbf{t}^*_{l,i}} \right)_{i=1\ldots|V|}$$

In $\mathbf{p}_k$, we have an entry for each vertex v_i representing the tendency that v_i will actually move to cluster C_k. This tendency is determined as the preference of cluster C_k for this object, weighted by the actual quality gained by v_i, and normalized by the overall tendency of v_i for all clusters. While a decision for the best cluster-vertex pair solely based on the gain matrix $\mathbf{G}_k$ was a too optimistic simplification of the problem at hand, a weighting with the tendency vector $\mathbf{p}_k$ allows a more realistic assessment. An entry $\mathbf{j}_{k,i}$ of our indicator vector $\mathbf{j}_k$ for cluster C_k describes the expected quality gain if the vertex v_i is transferred to cluster C_k.

$$\mathbf{j}_k = \mathbf{G}_k \cdot \mathbf{p}_k$$

Since the vertex-cluster pair with the highest entry $j_{k,i}$ is expected to be most beneficial with respect to later reassignments, we reassign v_i to cluster C_k (Algorithm 11.4 line 4). If the quality of the clustering has not improved, we stop our clustering-update (line 8), else we search for the next promising, so far unconsidered vertex to be reassigned (line 3).

Since the eigendecomposition for determining the clusters' preference vectors $\mathbf{t}_k$ is the computational bottleneck of this update function, we try to decrease effort and frequency of recomputing $\mathbf{t}_k$. Instead of recalculating all vectors $\mathbf{t}_k$ after assigning vertex v_i to cluster C_l, we directly update the overall preferences

$\mathbf{p}_{k,i}$, such that $\mathbf{p}_{l,i} = 1$ and all other probabilities $\mathbf{p}_{k,i} = 0$ for $k \neq l$ and recompute all $\mathbf{j}_k$ vectors before continuing the update step (line 7). A further enhancement concerns the size of the gain matrices $\mathbf{G}_k$ for each cluster. Since the reassignment of a vertex to a non-adjacent cluster can only decrease the cluster's modularity, we can restrict our computations to pairs (v_i, C_k) such that $v_i \in \cup_{v_j \in C_k} N(v_j)$, i.e., v_i is connected to at least one vertex from C_k. Further efficiency improvements like the exploration of the gain matrices' sparsity for the eigendecomposition are left for future work.

11.5 Experiments

In this section, we evaluate the effectiveness of our subspace modularity model and the performance of the different algorithms for its optimization described in Section 11.4 using synthetic and real-world data.

Experimental setup. We compare our subspace modularity measure Q_D to a 'full-space modularity' variant (Q_F) which simply sums up the edge weights of all dimensions and computes the modularity on the resulting graph. Our SuMo algorithm is compared to the adapted algorithms described in Section 11.4: the local reassignment function according to [BGLL08] (denoted by **LR**), cf. Algorithm 11.2 and the hierarchical clustering (denoted by **H**), cf. Algorithm 11.3. For both approaches, we apply the full-space objective function Q_F as well as the subspace modularity Q_D. As further competitor, we choose the popular approach of Newman [New06], which works on just a single weight and is not easily transferable to multi-dimensional weights. We therefore apply it as full-space variant with just a single weight as sum of the weights of all dimensions. All experiments were conducted on 2.33 GHz Intel Xeon CPUs with Java6 64-bit. We provide all used datasets on our website. For the experiments on synthetic data, we compare the detected results to the ground truth using the NMI measure (Normalized Mutual Information). As for the real-world datasets no ground truth is available, we can only compare key characteristics of the clustering results such as the achieved modularity scores, as well as the number K of detected clusters, and the runtime. Furthermore, we compute the NMI value of the results of SuMo to those of all other approaches, thereby evaluating the similarity of the results. Please note that the NMI does not correspond to a clustering quality here. Low NMI values indicate, however, that SuMo produces novel clustering results that cannot already be detected using the other objective functions or approaches.

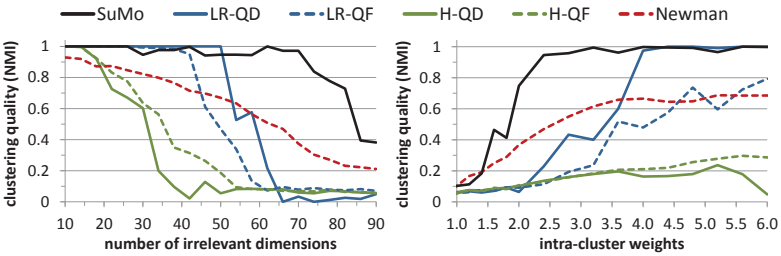


Figure 11.3: Quality vs. irrelevant dim. Figure 11.4: Quality vs. intra-weights

Experiments on synthetic data. We start by analyzing the different approaches in combination with each objective function for varying characteristics of the input datasets. Therefore, we generated a series of synthetic datasets. Each generated graph is 15-dimensional and consists of 4 clusters, each containing 30 to 50 vertices, and having 2 to 3 relevant dimensions.

In Fig. 11.3, we increase the number of irrelevant dimensions. Newman's approach behaves as expected for the full-space scenario and shows decreasing quality scores. LR-QF shows better results for few irrelevant dimensions and its robustness can be improved by the subspace modularity Q_D. For the H-algorithm, using Q_D has no noteworthy positive affect, also confirmed by the other experiments. Since hierarchical algorithms explicitly relinquish any backtracking, mistakes in the first iterations have a massive impact on the later clustering decisions. This counteracts with the Q^α idea of iteratively converging to the Q_D modularity. SuMo clearly proves to be more robust against irrelevant dimensions and can exploit the Q_D modularity more effectively than the other algorithms. If optimized successfully, the subspace modularity massively diminishes the negative effect of irrelevant dimensions.

In Fig. 11.4, we examine the impact of the actual weights on the clustering results. While for intra-cluster edges irrelevant dimensions have an average weight of 1, we vary the average weight of relevant dimensions from 1.0 to 6.0 (before normalization). Inter-cluster edges have an average weight of 1 for all dimensions. This experiment shows that SuMo is best capable of capturing the clustering structure if it is only weakly indicated by the weight-distribution. Also the LR-approach benefits from the subspace modularity Q_D. Full-space approaches only detect meaningful clusters if the intra-clusters draw a glaring picture.

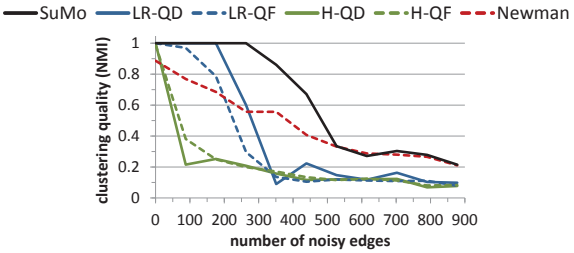


Figure 11.5: Quality vs. noise

For the experiment shown in Fig. 11.5, we increase the amount of noisy edges, which affects the graph structure and the weights as well. Again, we observe that LR performs better by using Q_D than with its full-space counterpart. With SuMo we can observe that the concentration on important information though subspace clustering helps balancing off the negative effects of noise.

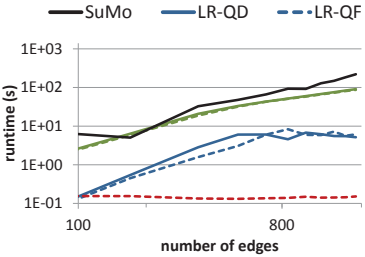


Figure 11.6: Runtime vs. # edges

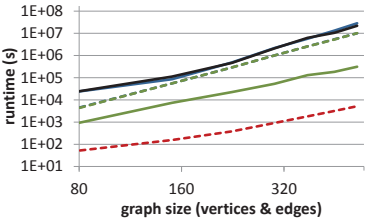


Figure 11.7: Runtime scalability

Although our focus is on evaluating the cluster quality, we briefly discuss the methods' efficiency. In Fig. 11.6, we increase the number of edges of the graph. All algorithms scale linearly (note the logarithmic scaling of both axes) and we see that the application of Q_D only marginally affects the runtime. While SuMo and the hierarchical algorithm show a similar runtime, the algorithm of [BGLL08] and Newman clearly outperform them both. In Fig. 11.7, we increase the number of nodes in a graph, which is accompanied by a quadratic increase of the number of edges (logarithmic scaling of both axes). Accordingly, the runtimes of all algorithms increase super-linearly. Newman is the overall most efficient algorithm. However, only solving the full-space clustering problem.

	NMI	Q_D	Q_F	K	runtime
SuMo (Q_D)	1	0.69	0.22	22	54,594 s
LR (Q_D)	0.49	0.69	0.20	4	1,057 s
LR (Q_F)	0.42	0.59	0.52	31	235 s
H (Q_D)	0.45	0.67	0.50	88	20,039 s
H (Q_F)	0.41	0.64	0.53	84	15,564 s
Newman	0.33	0.58	0.46	39	25 s

Table 11.1: Clustering results on IMDB

NMI	Q_D	Q_F	K	runtime
1	0.76	0.70	25	7,449 s
0.77	0.77	0.73	18	115 s
0.79	0.77	0.75	40	67 s
0.8	0.78	0.74	50	8,467 s
0.79	0.78	0.74	48	7,518 s
0.72	0.73	0.70	54	34 s

Table 11.2: Clustering results on arXiv

Experiments on real-world data. Our first real-world dataset is an extract of the IMDb movie database (www.imdb.com). The vertices represent movies produced in USA, Canada, UK, or Germany, which are connected to each other if they share actors or if there exists a reference (e.g., spoof or follow up) between them. The edge weight dimensions represent 21 movie genres. Overall, the network contains 862 nodes and 4388 edges.

For the results of this dataset, shown in Table 11.1, we observe that all approaches using Q_D as their objective function obtain similarly good Q_D values but lead to highly disagreeing clustering results indicated by the low NMI-values of maximum 0.49. While the results of the Q_D-optimizing algorithms achieve the highest Q_D scores, the equivalent scores for the modularity in the full-space are rather low. This disagreement of the measures indicates a notable fraction of irrelevant dimensions, correctly treated by the algorithms. Contrarily, approaches operating in the full-space show similar values for Q_D and Q_F. A further advantage of our subspace modularity is its ability to reveal additional information about the relevant dimensions which allows a semantic interpretation of the clusters. SuMo, e.g., detects a cluster with relevant dimensions "Thriller", "Mystery", and "Horror" containing movies like "The Ninth Gate" or "Lady in White". Another cluster's relevant dimensions are "Biography", "History", and "War". It contains 18 movies such as "The Times of Harvey Milk" or "Winter Soldier".

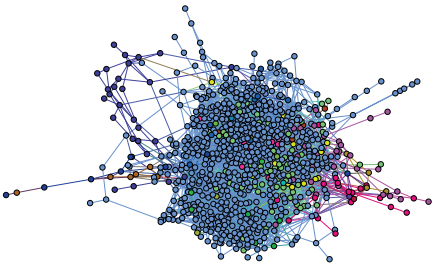


Figure 11.8: Clusters in IMDb

Our second real-world dataset, is extracted from the arXiv online archive (www.arxiv.org). In this network, each vertex represents a paper and each edge represents a citation between two papers. The edges are annotated with a 30-

dimensional weight vector, where each dimension represents one of the top 30 keywords. The weight in each dimension denotes the minimum number of occurrences of the keyword in the papers' abstracts. Overall, the dataset contains 856 vertices and 2660 edges.

For this dataset, all approaches reach quite similar values for all the modularity measures (cf. Table 11.2). To understand this behavior, differing to the IMDB dataset, we have to take a look at the structure of the graphs. In Fig. 11.8 and 11.9 we visualized both graphs using the ForceAtlas2 algorithm on the Gephi platform (www.gephi.org), which arranges the vertices based on the graph structure. Vertices having the same color, share a cluster in the result of SuMo. We observe that the arXiv graph is quite sparse and the clusters fit the graph structure well. As clusters can already be detected looking solely at the graph structure, the additional consideration of edge weight subspaces is not advantageous. In contrast, the IMDB graph is very dense and it is hardly possible to detect clusters by looking just at the graph structure. In this case, detecting clusters considering just the relevant dimensions by using the subspace modularity leads to a clearer clustering structure than using the traditional modularity. For both graphs, full-space clustering leads to a higher number of clusters than subspace clustering, confirming the theory that vertices rarely show strong connections for all dimensions.

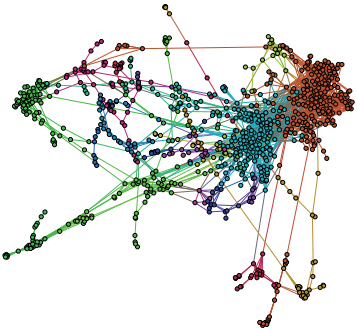


Figure 11.9: Clusters in arXiv

11.6 Conclusion

For the presented approach of this chapter, we extended the well-known modularity measure to handle graphs with multi-dimensional edge weights by following the principles of subspace clustering. We have shown how our modularity extension can be adopted by some of the existing modularity optimization approaches and proposed the clustering algorithm SuMo for more effectively clustering networks based on the subspace modularity. The efficiency of optimizing a clustering regarding the subspace modularity is so far not satisfactory and, thus, still an open research problem.

12

Evaluation of Graph Techniques for Alternative Clustering

THE graph-based framework that we presented in Section 9.3 theoretically complies with the six challenges for iterative multi-view clustering defined in Chapter 9. In this chapter, we want to examine in a small study whether this framework also practically proves to be useful to iteratively search for multiple clustering alternatives. As instantiation for the actual graph clustering algorithms, we will examine the approaches presented in the two previous chapters: the algorithm SSCG of Chapter 10 which uses a spectral clustering approach, and the two promising approaches of Chapter 11, LR-QD (local reassignment function) and SuMo (eigendecomposition of the gain matrix) optimizing the novel subspace modularity. Since for the subspace modularity the hierarchical clustering procedure presented in Chapter 11 performed unsatisfyingly in the subspace scenario, we will not consider it here.

In order to evaluate different characteristics of the algorithms and to be able to compare results against a ground truth, we will focus on synthetic data. Based on the generative model SMVC presented for multi-view data in Chapter 8, we generate a relatively simple dataset, where the clusters are not cluttered together but are well separated in their respective subspaces. Since we start our evaluation with the scenario where no prior information about previous clusterings is available and therefore have to consider a complete graph, we restrict our evaluation to just a small dataset consisting of 100 objects and 15 dimensions. The number of views varies for each experiment. Each view contains 3 clusters and the views' relevant subspaces are disjoint. The quality is assessed based on the $E4SC$ measure (Chapter 13), a symmetric, subspace aware variant of the popular F1 measure. We refrain from evaluating the subspaces and just concentrate on the object groupings (for clarity we rename the measure to 'E4FC'). For all quality experiments, we average the results over ten executions on different datasets.

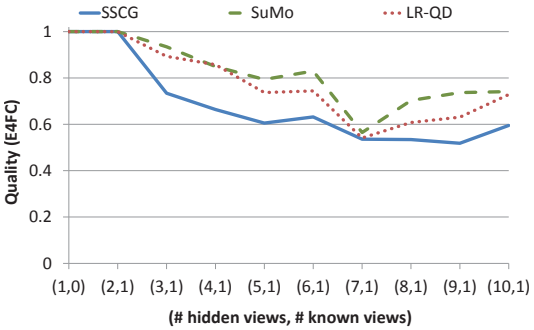


Figure 12.1: Quality for a varying number of hidden views and one known view

In the first experiment in Fig. 12.1, we evaluate how the different algorithms perform for a different number of hidden clustering views when only a single clustering is provided as prior knowledge. For each algorithm, we execute one iteration to find the next hidden view, which is compared against the best matching ground truth view based on the E4FC measure. Overall, we observe a slightly better clustering quality for the approaches optimizing the subspace modularity, SuMo and LR-QD, where our new approach SuMo performs best. Since, in contrast to the SSCG approach, these algorithms are not provided with the correct number of clusters and, thus, have fewer information available, this is an interesting result. For just a small number of hidden views, the graph based framework manages to recover a novel hidden clustering to a large extent. However, with an increasing number of hidden views, the agreement between the newly discovered clustering and a hidden clustering drops. Given too many hidden clusterings in the data, the graph clustering algorithms are not able to focus on a single hidden clustering anymore, but merge the information of several clusterings. Of course, the appearance of six or more strong clustering views in a single dataset is unlikely to be the standard scenario. Nonetheless, the observation that partitioning graph clustering methods struggle with the multi-view scenario indicates a fruitful future research direction of transferring the multi-view clustering principles to the graph clustering problem.

In the next experiment (cf. Fig. 12.2), we want to examine whether the algorithms can successfully integrate the knowledge of multiple provided clusterings, i.e., whether the number of given clusterings has an influence on the clustering result. We compare four scenarios: we try to find a single novel clustering if no prior information is available (“0 known views”), we try to find a single novel

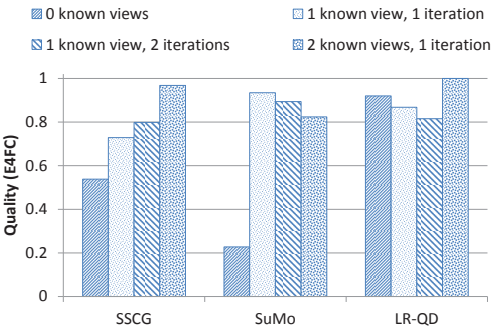


Figure 12.2: Quality for a varying number of known views for 3 hidden views

clustering if we know about a single ground truth clustering ("1 known view, 1 iteration"), we try to find the third hidden clustering if we just know a single ground truth clustering, i.e., we use the clustering detected in the first iteration as additional prior information for the second clustering iteration ("1 known view, 2 iterations"), and the last scenario where we provide two ground truth clusterings and try to discover the third hidden clustering ("2 known views, 1 iteration"). SSCG is the only algorithm that behaves as expected. The more prior information SSCG has, the better is the quality of the newly detected clustering. SSCG also benefits from a higher quality of the given clusterings, indicated by a better clustering quality if two ground truth clusterings are given compared to the third scenario of one ground truth clustering and one clustering that has been determined by SSCG itself. This second observation also holds for the LR-QD algorithm. For SuMo, we observe two interesting effects. First, the clustering quality for no prior information is very bad for three hidden views. Since the full graph does not provide any support for the clustering, SuMo is conflicted with the different views in the feature space and detects several views simultaneously, resulting in too many clusters. With prior information, SuMo discovers the hidden clustering structure significantly better. Second, since the quality of the first detected alternative clustering is already very high, replacing it by a ground truth clustering does not gain a high benefit for the second iteration.

In summary, if no prior information is available SSCG or LR-QD perform better than SuMo. For little prior information the subspace modularity based approaches SuMo and LR-QD perform better than SSCG. However, all approaches struggle with the conflicting clustering views hidden in the data and provide potential for future improvements in the multi-view clustering setting.

Part V

Evaluating and Visualizing Alternative Clustering Solutions in Subspace Projections

Measure what is measurable, and make measurable what is not so.
Galileo Galilei

13

External Evaluation Measures for Subspace Clustering

In young research areas where no common objective evaluation measures are available, researchers are usually unable to provide a fair and comparable quality assessment of their newly developed methods. Typically, publications glorify the high quality of one approach only justified by an arbitrary evaluation measure. However, such conclusions can only be drawn if the evaluation measures themselves are fully understood.

In this chapter, we provide the basis for a systematic evaluation in the emerging research area of subspace clustering. We formalize general quality criteria for subspace clustering measures not yet addressed in the literature. We compare the existing external evaluation methods based on these criteria and pinpoint limitations. We propose a novel external evaluation measure which meets the requirements in form of quality properties. In thorough experiments, we empirically show characteristic properties of evaluation measures. Overall, we provide a set of evaluation measures that fulfill the general quality criteria as recommendation for future evaluations.

13.1 Introduction

For knowledge discovery in databases, fair and comparable evaluation of detected patterns is of major importance. For a thorough evaluation of mining techniques, it is essential to have objective methods that measure the quality of data mining results. In contrast to the subjective quality assessment by domain experts, these measures should provide an objective and comparable evaluation. This evaluation is important for quality assessment of novel methods versus competing approaches but also for knowledge extraction based on the detected patterns. Evaluation completes the knowledge discovery process by providing more insights than a mere listing of patterns.

In this chapter, we focus on evaluation measures for subspace clustering techniques [PHL04, KKZ09]. In general, *subspace clustering* and *projected clustering* aim at the detection of clusters in arbitrary subspace projections. While traditional clustering searches for clusters based on object similarity using all available attributes (full-space), subspace clustering considers object similarity in any subset of the given attributes (subspaces). So far, only few measures were developed specifically for subspace clustering. Instead, researchers borrowed measures from other areas, such as information retrieval or classification without discussing their applicability and characteristics for subspace clustering. Thus, some measures may not be appropriate for subspace cluster evaluation. Furthermore, the differing use of measures leads to incomparable results. As illustrated in Fig. 13.1, comparing the hidden clusters (ground truth) and the detected clusters with two of these measures might yield contradicting results. Even for a single measure, users do not know how to interpret the results because they are not aware of its characteristics. Different evaluation measures, usually focus on different aspects of a clustering and hardly any of them allows a holistic evaluation.

In this chapter, we bridge the gap between individual evaluation measures. Besides evaluation challenges inherited by traditional clustering, we highlight specific core requirements for a systematic evaluation of subspace clustering results. Based on a given ground truth, traditional clustering measures evaluate the purity of clusters and the detection of all clustered objects. For subspace clustering we have further challenges: First, a high quality subspace clustering should also detect the correct subspaces in which objects are grouped. Second, subspace clusters might be reported redundantly in several subspaces. Last, objects may be part of multiple valid clusters.

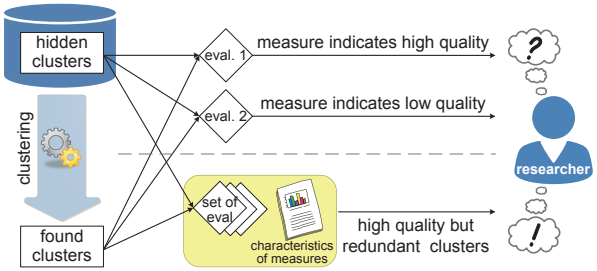


Figure 13.1: Enhanced evaluation by insights into characteristics of evaluation measures [conflicting evaluation (top), meaningful interpretation (bottom)]

As key contributions, we take a systematic approach to characterize the main *quality requirements for subspace clustering*. We formalize all of these properties and we provide an *analysis of evaluation measures* used in recent publications [MSE06, PM06, AKMS07a, MS08, AKMS08b, MAG+09b, MGAS09, MZK+09]. In addition, we propose an *enhanced evaluation measure* which meets these requirements. Based on our systematic comparison, we provide a *recommendation of measures* to be used in future evaluations. In conjunction with the *derived characteristics* in this chapter, these measures can be used not only for evaluation but also for further knowledge extraction. Knowing about the characteristics of the measures, the evaluation allows to reflect on the reasons for poor results (Fig. 13.1 (bottom)). This knowledge can be helpful for better parametrization or for improvement of the data mining algorithm itself.

13.2 Subspace Cluster Evaluation

Evaluation of clustering as unsupervised learning is challenging since the "correct" result is usually unknown. Several evaluation types have been proposed.

Evaluation Types Evaluation based on domain experts is one possible type, used in application oriented evaluations. Here, domain experts are consulted to manually evaluate each cluster. This evaluation provides more insight into the detected clusters but it is subjective and does not yield comparable results on benchmark data. Furthermore, it can only be applied for very small result sets. As a second evaluation type, internal evaluation measures are defined based on properties of the cluster definition, e.g., the compactness of clusters

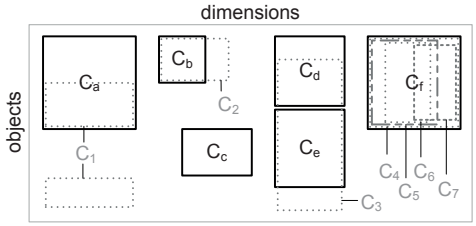
(cf. k-means [LLX+10]). Such measures only reflect the relative adherence to the underlying cluster definition. They are, thus, typically used for those clustering paradigms trying to optimize a task specific objective function. For most clustering paradigms including subspace clustering such a general objective function is not defined, and thus, no internal measure is equally meaningful for all methods. Clustering methods adhering to different cluster definitions cannot be fairly evaluated w.r.t. a single internal measure. As a third type, external evaluation measures are used (e.g., for k-means [WXC09]). They assume a ground truth, as provided by synthetic data or labeled data. External measures compare the detected clusters with this given ground truth, providing an objective quality assessment, independent of the cluster definition. In this work, we focus on external evaluation measures for subspace clustering. Before discussing the novel requirements induced by subspace clustering, we review the general idea of external evaluation measures.

The "All and Only" Quality Criterion External evaluation measures compare a given ground truth (ideal clustering) with the detected result set of found clusters. Intuitively, a measure should provide high quality values for a clustering that detects *all* hidden clusters, but also detects *only* the hidden clusters. This *all and only* property applies to several aspects in the evaluation of subspace clusters. We distinguish between the cluster level (single cluster) and the clustering level (overall set of clusters): First, on the cluster level, each found cluster should contain *all and only* objects of a single hidden cluster. Furthermore, each found subspace cluster should be detected in *all and only* the dimensions of the hidden subspace cluster. And second, on the clustering level, the overall set of detected clusters should contain *all and only* the hidden clusters.

Evaluation characteristics We first introduce some basic notions for subspace cluster evaluation, before presenting the quality requirements that each measure should fulfill. An external measure evaluates the subspace clustering result that contains a set of subspace clusters, each representing a group of objects in a subset of the dimensions.

Definition 13.1 *Subspace clustering result*
Given a set of dimensions Dim and a database DB, a subspace cluster $C = (O, S)$ is a set of objects $O \subseteq DB$ along with a set of relevant dimensions $S \subseteq Dim$. A subspace clustering result Res is a set of subspace clusters $Res = \{C_1, \ldots, C_k\}$ with C_i being a subspace cluster.

In Fig. 13.2, two exemplary subspace clusterings are illustrated. The x-axis denotes the dimensions and the y-axis the objects of the database. Each subspace cluster covers a specific set of objects and dimensions.



Subspace clustering 1: $\{C_1, C_2, \ldots, C_7\}$
Subspace clustering 2: $\{C_a, C_b, \ldots, C_f\}$

Figure 13.2: Two exemplary subspace clusterings

External evaluation measures determine the quality of a clustering w.r.t. a ground truth. This ground truth represents a gold standard that should be recovered by the subspace clustering algorithms to the greatest extent.

Definition 13.2 *Ground truth*
The ground truth $Ground$ is a subspace clustering representing the perfect result.

In Fig. 13.2, we assume the ground truth to be given by $Ground = \{C_a, \ldots, C_f\}$ and the other clustering to be determined by a clustering algorithm. As indicated in Fig. 13.2, an object can belong to several subspace clusters. Similarly, the relevant dimensions of clusters can overlap. Hence, an object can be part of several clusters in a single dimension in the ground truth or in the clustering result, e.g., the ones of C_4 and C_5. Thus, a mandatory requirement for each measure is to handle overlapping subspace clusters. We denote this criterion as *overlap applicable*.

An evaluation measure for subspace clustering can formally be defined by:

Definition 13.3 *Evaluation measure*
Given a set of dimensions Dim, a database DB, and the ground truth $Ground$ of this dataset; an evaluation measure is a function M:

$$M : \mathcal{P}(Clus) \times \mathcal{P}(Clus) \to \mathbb{R}$$

where $Clus = \{(O, S) \mid O \subseteq DB, S \subseteq Dim\}$ is the set of all possible subspace clusters. The quality of a clustering Res w.r.t. the ground truth is: $M(Ground, Res)$.

In general, an external measure can be used as a similarity measure between two arbitrary clusterings. W.l.o.g., in our work all measures are normalized between 0 and 1, where 1 indicates the perfect quality, e.g., $M(Ground, Ground) = 1$ holds. Any errors in the result should be reflected in the value of M and, hence, an optimal value should only be achieved for identical clusterings. As depicted in Fig. 13.2, the clusters themselves can differ, i.e., the hidden clusters are not exactly recovered (C_b vs. C_2); but also the overall set of clusters can differ, i.e., the clusterings disagree (C_c). Overall, we discuss specific characteristics on the cluster level and on the clustering level. We start with the cluster level.

Object awareness As in traditional clustering, we want to identify the correct object groupings of the hidden clusters. The found clusters should not mix several hidden clusters or obfuscate a hidden cluster by other objects, since the *purity* of a cluster is crucial. The cluster C_2 in Fig. 13.2 has perfect purity w.r.t. C_b regarding the objects, while the cluster C_3 mixes several hidden clusters and noise objects. Moreover, for a correct object grouping, it is also important to identify as many objects as possible of the hidden cluster, not just a few, as does the cluster C_1 w.r.t. C_a. Overall, for a good detection, it is mandatory to group *all and only* the objects of the hidden cluster. If this is not fulfilled by the clustering result Res, a measure M should determine a lower quality. We denote this property of a measure as *object awareness*.

Subspace awareness For subspace clusters, the set of relevant dimensions constitutes a major part of its information content. It is, therefore, important to identify the correct object group and, *at the same time*, the correct relevant dimensions. It is an indication of poor quality to find the hidden object group but in a totally different subspace. Consequently, we want to identify *all and only* the relevant dimensions of a subspace cluster of the ground truth. In Fig. 13.2, the cluster C_2 does not perfectly reflect the relevant dimensions of C_b. A measure fulfills the *subspace awareness* criterion if it penalizes false or missing relevant dimensions.

Redundancy awareness Both previous criteria are relevant for determining the quality of *single* subspace clusters. The following criteria will consider the clustering level. In subspace clustering, we analyze subspace projections of the data. For any subspace cluster definition fulfilling the anti-monotonicity criterion, all exponentially many subspace projections of a valid cluster are valid as well. A set of clusters sharing nearly all objects and relevant dimensions, however, induces

redundancy and, therefore, obscures the true clustering result. A measure should penalize clustering results that identify one ground truth cluster several times. Besides the true hidden cluster C_f/C_4, in Fig. 13.2 several redundant clusters are generated (C_5, C_6, C_7). In traditional clustering, redundancy does not occur due to full-space clustering. In subspace clustering, however, several approaches suffer from this phenomenon [AGGR98, NGC01, KKK04]. Evaluation measures for subspace clustering have to take into account that a redundancy polluted clustering is not the perfect clustering. Adding further clusters not represented by the ground truth must lead to a lower quality. In extreme words: Simply generating all possible clusters must not yield the perfect quality. A measure accounting for this criterion is *redundancy aware*.

Identification awareness Respectively, it is not optimal to miss some clusters of the ground truth. In Fig. 13.2, the cluster C_c is not identified at all; a measure should not determine perfect quality. For subspace clustering, this property is a challenge which cannot be adapted trivially from traditional clustering. While in traditional clustering, each object belongs to just one cluster, in subspace clustering, objects can belong to several clusters due to their relevant dimensions. Missing clusters in traditional clustering can simply be identified by a non coverage of some objects. However, in subspace clustering, all objects could be covered by the result, even if not all clusters are identified. This problem appears, e.g., in partitioning approaches that only detect disjoint subspace clusters [AWY+99, AY00b, PJAM02, YM03]. Overall, to be *identification aware*, a measure has to decrease the quality for every missing cluster of the ground truth.

Summarizing, we introduce four criteria that evaluation measures for subspace clustering have to fulfill: object, subspace, redundancy, and identification awareness.

13.3 Evaluation Measures

In this section, we examine existing evaluation measures with regard to the four criteria. Only if a measure responds to all four respective clustering variations, substantial conclusions can be drawn out of its quality assessment. For measures that ignore at least one criterion, a low quality value still indicates a bad clustering solution. A high quality value, contrarily, does not necessarily indicate a good clustering result. Clearly, a measure where neither low nor high quality values

allow any conclusions is inappropriate. This is the case for those, that do not handle mandatory requirements of subspace clustering such as handling overlaps. An example is the $Entropy$ measure [AKMS07a, MGAS09]. For a ground truth with overlapping clusters, the entropy will always indicate a quality below optimal, even if we compare the ground truth against itself. A clustering that equals the ground truth should, however, always have optimal quality results.

In the following Section 13.3.1, we only consider those measures that are overlap applicable. To the best of our knowledge, we include all evaluation measures in our comparison that are used in recent subspace clustering publications. For these, we will examine the sensitivity w.r.t. our 4 criteria. As a result, we will get that none of the existing measures deals fairly with all criteria. We, therefore, propose a novel, simple quality measure for subspace clustering in Section 13.3.2.

13.3.1 Analysis of Existing Measures

For our analysis, we assume Res to be the set of found clusters and $Ground$ to be the ground truth. The objects of a cluster C are denoted with $O(C)$ and the set of relevant dimensions of C with $S(C)$ respectively.

F1 measures

One method for the evaluation of clustering results is the F1-measure. F1 formalizes the requirement that clusters in Res should represent the clusters in $Ground$. That is, a cluster $C_r \in Res$ should, on the one hand, have many objects in common with one of the hidden clusters $C_g \in Ground$, but, on the other hand, it should contain as few objects as possible that are not in this particular hidden cluster. These two constraints can be formalized by the terms precision and recall and represent the all and only constraint of object awareness.

$$recall(C_r, C_g) = \frac{|O(C_r) \cap O(C_g)|}{|O(C_g)|} = precision(C_g, C_r)$$

The F1-measure evaluates the matching of two clusters as the harmonic mean of precision and recall.

$$F1(C_r, C_g) = \frac{2 \cdot recall(C_r, C_g) \cdot precision(C_r, C_g)}{recall(C_r, C_g) + precision(C_r, C_g)}$$

Note that the relevant subspaces do not occur in the formal definition of F1, which is, therefore, not aware of subspaces. However, it is widely used in subspace clustering evaluation [MSE06, MS08, MZK+09, AKMS08b, MAG+09b]. For the overall matching of two clusterings P and Q the F1 measure is defined as:

$$F1^{Clus}(P,Q) = \frac{1}{|P|} \sum_{C_i \in P} \max_{C_j \in Q} \{F1(C_i, C_j)\}$$

Optimal quality is denoted by a value of 1, whereas 0 indicates the lowest quality. It is crucial that the function of $F1^{Clus}$ is not symmetric ($F1^{Clus}(P,Q) \neq F1^{Clus}(Q,P)$), even though this has not been discussed in the literature yet. The sum only iterates over clusters in P and thus clusters in Q are only considered if they match at least one cluster in P best. Besides these matches, clusters in Q have no influence on the evaluation result at all. The measure is used as

$$M^{F1-R}(Ground, Res) := F1^{Clus}(Ground, Res)$$

in [AGAV09], where Res is the clustering that is considered only partially for the quality evaluation. Thus, this definition is not redundancy aware, since it is not capable of detecting the presence of false clusters. That is, for all clusterings $Res \supseteq Ground$, the quality result will always be optimal: $M^{F1-R}(Ground, Res) = 1$. Thus, only the obtained recall w.r.t. the clusters in $Ground$ is assessed; hence, the naming "$F1$-Recall" ($F1$-R).

In [MSE06, MS08, MZK+09], results are evaluated by the counterpart definition

$$M^{F1-P}(Ground, Res) := F1^{Clus}(Res, Ground)$$

In this case, the ground truth has only limited influence on the quality assessment of F1. Therefore, the presented definition of F1 is not identification aware, as all clusterings $Res \subseteq Ground$ will always have the perfect quality outcome of $M^{F1-P}(Ground, Res) = 1$. Since only the obtained precision w.r.t. the clusters in $Ground$ is assessed, we chose the naming "$F1$-Precision" ($F1$-P).

In [AKMS08b, MAG+09b, MGAS09], a third definition was introduced, where clusters in Res are merged if their best matching cluster in $Ground$ is identical. The size of the resulting clustering Res' is thus adjusted to the size of $Ground$. For each cluster $\overline{C} \in Ground$, we get a new cluster $C' \in Res'$, such that:

$$O(C') := \bigcup_{C \in Res} \left\{ O(C) \mid \overline{C} = \operatorname*{argmax}_{C_i \in Ground} \left\{ \frac{|O(C) \cap O(C_i)|}{|O(C_i)|} \right\} \right\}$$

A solution Res is evaluated based on the ground truth and the novel result Res':

$$M^{F1-Merge}(Ground, Res) := F1^{Clus}(Ground, Res')$$

Due to the merging of found clusters, this measure does not detect whether found clusters split hidden clusters and is, thus, not object aware.

Accuracy

Another quality assessment is realized by the Accuracy measure [MAG+09b, BZ07, MGAS09]. The basic idea is to predict the hidden clusters based on the found clusters. The more accurate the hidden clusters are predicted, the better the ground truth is generalized by the identified clusters. For prediction, the method of classification is used. As the training data, bitvectors are given that represent the membership of the objects in the hidden clusters $C_i \in Res$. That is, each object o induces a bitvector of length $k = |Res|$ where the ith entry is 1 if $o \in C_i$. Based on this training data, a decision tree classifier is built and the accuracy is determined (usually C4.5 with 10-fold cross validation).

Since classification accuracy depends on the training data, we can infer that impure clusters affect the quality of the result; the training data contains errors w.r.t. the ground truth. However, as a classifier tries to countervail these effects, the object awareness of the measure is questionable: Even if an object was assigned to some wrong clusters, the classifier could be able to predict the correct hidden clusters for the object. Thus, the measure indicates high quality, even in the presence of errors in the clustering result. Obviously, this measure is also not subspace aware, because only the object sets are used for training. If a cluster is completely missed, the classifier cannot assign the objects to this cluster. Thus, the identification awareness is fulfilled.

RNIA

[PM06] introduced the first measures that fulfill the subspace awareness criterion. The basic idea is to represent a cluster (O, S) as a single set T instead of a tuple. For this, each object $o_i \in O$ is not treated as the full-dimensional feature but for each dimension $d \in S$ an object $o_{i,d}$ is constructed. We denote these objects as micro-objects. Thus, a subspace cluster can be represented by its set of micro-objects

$$t(C) = \{o_{i,d} \mid o_i \in O(C) \wedge d \in S(C)\}$$

An x-dimensional cluster with y objects is represented by $x \cdot y$ micro-objects. Based on this representation, the RNIA (relative non-intersecting area) measure assesses whether the micro-objects of the ground truth are all and only covered by the clustering result. Formally, the union U of the micro-objects of both clusterings is determined and their intersection I is subtracted. The assumption is, that for a good clustering result, U and I are nearly identical. Overall,

$$M^{RNIA}(Ground, Res) := \frac{|U| - |I|}{|U|}$$

with $U = U(Ground) \cup U(Res)$, $I = U(Ground) \cap U(Res)$, and $U(P) = \left[\bigcup_{C \in P} t(C)\right]$. To handle overlapping clusters, [PM06] presents a method to adapt the union and intersection. We use this version in our experiments and we plot the value $1.0 - RNIA$ so that perfect quality corresponds to 1.0.

Obviously, the RNIA measure is subspace aware. The redundancy and identification awareness are also fulfilled, because errors w.r.t. these criteria have an influence on the union and intersection respectively. The drawback of RNIA is its lack of object awareness. The purity or recall of single clusters is not considered at all. RNIA simply checks whether a micro-object of the ground truth is also contained in Res and vice versa. If $U(Ground) = U(Res)$ holds, RNIA returns perfect quality; independent of how the single clusters behave. Splits or impurities of clusters remain undetected.

CE

The disadvantages of the RNIA measure were addressed by the CE (clustering error) measure [PM06]. The basic idea is to find a 1:1 mapping between the hidden and found clusters. Each cluster C_g of the ground truth is assigned to at most one cluster C_r of the result, and vice versa. For each mapped pair (C_g, C_r) the cardinality of their intersecting micro-objects is determined. Overall, only those 1:1 mappings are chosen that result in the highest total sum over all cardinalities. This sum is denoted as D_{max}. By replacing the intersection I within RNIA by D_{max}, we formally get the CE measure:

$$M^{CE}(Ground, Res) := \frac{|U| - D_{max}}{|U|}$$

We plot the values of $1.0 - CE$ such that perfect quality equals to 1. CE fulfills the same quality criteria as RNIA. The object awareness is still not completely fulfilled. On the one hand, the 1:1 mapping penalizes clusters which split up in

several smaller ones because the coverage of the cluster decreases. On the other hand, the impurity of clusters is still not considered; the intersection between two clusters is not influenced by additional, wrong objects. Thus, the object awareness is not adequately implemented.

13.3.2 The E4SC Evaluation Measure

The previously reviewed techniques have major drawbacks in at least one of the 4 awareness criteria. The gathered insights on subspace clustering allow us to define an external evaluation measure, that addresses all 4 criteria, for a holistic evaluation of subspace clusterings.

The terms of precision and recall assure the awareness of objects with full contentment. Thus, they build the basis of our new E4SC measure. By transforming subspace clusters to micro-object clusters, the object awareness is extended to dimensions. The definitions of recall and precision become:

$$recall_{SC}(C_r, C_g) = \frac{|t(C_r) \cap t(C_g)|}{|t(C_g)|} = precision_{SC}(C_g, C_r)$$

The harmonic mean of precision and recall now represents the all and only constraint for cluster objects as well as for relevant dimensions. On the cluster level, this measure meets all requirements for subspace cluster evaluation.

$$F1_{SC}(C_r, C_g) = \frac{2 \cdot recall_{SC}(C_r, C_g) \cdot precision_{SC}(C_r, C_g)}{recall_{SC}(C_r, C_g) + precision_{SC}(C_r, C_g)}$$

The extension $F1_{SC}^{Clus}(P, Q)$ of this definition for the clustering level leads to

$$F1_{SC}^{Clus}(P, Q) = \frac{1}{|P|} \sum_{C_i \in P} \max_{C_j \in Q} \{F1_{SC}(C_i, C_j)\}$$

This formula exhibits a non-symmetry that we utilize for enhanced quality assessment. The non-symmetry of $F1_{SC}^{Clus}$ implicates a precision and recall relation itself – though on the cluster*ing* level. $F1_{SC}^{Clus}(Ground,Res)$ evaluates how well all of the hidden clusters were found; it can be denoted as the *recall of the clustering*. Contrarily, $F1_{SC}^{Clus}(Res, Ground)$ evaluates how well each of the found clusters represents one of the hidden clusters; thus, it can be seen as *precision of the clustering*. The combination of these derived precision and recall values by a harmonic mean represents the all and only constraint on the clustering level.

$$M^{E4SC}(P,Q) := \frac{2 \cdot F1_{SC}^{Clus}(P,Q) \cdot F1_{SC}^{Clus}(Q,P)}{F1_{SC}^{Clus}(P,Q) + F1_{SC}^{Clus}(Q,P)}$$

The novel measure of E4SC successfully adopts the idea of F1 for subspace clustering. The central idea of precision and recall has been transferred to the *level of subspace clusters* and by means of recurring averaging has also upgraded F1 to the *level of clustering*. E4SC stands out due to the complete consideration of all 4 criteria. Object awareness is realized by using precision and recall on cluster level. A maximal quality result thus reports pure and complete clusters in Res compared to $Ground$. As dimensions are treated as micro-objects, the same holds for subspace awareness. Through the harmonic mean of $F1_{SC}^{Clus}(P,Q)$ and $F1_{SC}^{Clus}(Q,P)$, E4SC is able to consider lower quality due to redundant or missing identification of clusters. The recall $F1_{SC}^{Clus}(Ground, Res)$ of the clustering decreases if one cluster in $Ground$ is not or insufficiently found by Res. The precision $F1_{SC}^{Clus}(Res, Ground)$ is low if clusters in Res are unrelated to the clusters in $Ground$. An optimal quality result, thus, also reports a pure and complete clustering Res with regard to $Ground$. The maximal quality value of $M^{E4SC}(Ground, Res) = 1$ indicates an optimal clustering w.r.t. all characteristics of subspace clustering.

Furthermore, E4SC fulfills the following useful properties: symmetry, non-negativity and identity of indiscernibles. The symmetry property $M^{E4SC}(P,Q) = M^{E4SC}(Q,P)$ for all clusterings Q and P is valid by design. Since precision and recall values lie within the range $[0,1]$, the harmonic mean does so too. Thus, E4SC fulfills the non-negativity and, more importantly, for all clusterings P and Q, we get $M^{E4SC}(P,Q) \in [0,1]$. At last, we prove the identity of indiscernibles, i.e., $P = Q \Leftrightarrow M^{E4SC}(P,Q) = 1$. This property is especially important since the perfect quality is only achieved if the two clusterings are identical. Any error in the clustering result, e.g., splits, redundancy, or inclusion of noise, leads to a decrease of the E4SC value.

Proof 13.1 $\Rightarrow$: *It holds that* $recall_{SC}(C,C) = 1 = precision_{SC}(C,C)$ *since* $t(C) \cap t(C) = t(C)$ *and therefore also* $F1_{SC}(C,C) = 1$. *If* $P = Q$, *we have* $\forall C_i \in P\ \exists C_j \in Q:\ C_i = C_j$ *and since* $F1_{SC}(C_r, C_g) \leq 1$ *we get*

$$\frac{1}{|P|} \sum_{C_i \in P} \max_{C_j \in Q} \{F1_{SC}(C_i, C_j)\} = \frac{1}{|P|} \sum_{C_i \in P} 1 = 1$$

Thus, $F1_{SC}^{Clus}(P,P) = 1$ *and hence* $M^{E4SC}(P,P) = 1$.

$\Leftarrow$: *Assuming* $P \neq Q$. *W.l.o.g. there exists* $C \in P$ *and* $C \notin Q$. *It holds that* $\forall C_j \in Q: \ F1_{SC}(C, C_j) < 1$ *since either* $recall_{SC}(C, C_j) < 1$ *or* $precision_{SC}(C, C_j) < 1$. *Otherwise* $C \in Q$ *would hold. Thus*

$$F1_{SC}^{Clus}(P, Q) \leq \frac{1}{|P|} \left(|P \backslash \{C\}| + \max_{C_j \in Q} \{F1_{SC}(C, C_j)\} \right)$$

$$< \frac{1}{|P|} \left(|P \backslash \{C\}| + 1 \right) = 1$$

and hence the harmonic mean $M^{E4SC}(P, Q) < 1$ *for* $P \neq Q$.

13.4 Experiments

In our experiments, we highlight the characteristics of the evaluation measures w.r.t. errors in the clustering result. While all measures provide perfect quality for a perfect clustering, they show different behaviors on clustering errors. Given a ground truth $Ground$ of hidden subspace clusters and a clustering result Res of found subspace clusters, we compare the measures' performance. In order to understand all properties of the measures under consideration, we first study them for different evaluation scenarios with synthetic data, where we are able to vary the degree of each error separately as illustrated in Fig. 13.3 (right). To study different characteristics, we emphasize a different error in each evaluation scenario. Each scenario is motivated by general errors produced by different clustering approaches. The resulting insights subsequently allow us to discuss some evaluations of real cluster algorithms on real world datasets. These experiments will reveal the dependency of the assessment of the vanquishing algorithm on the evaluation measure.

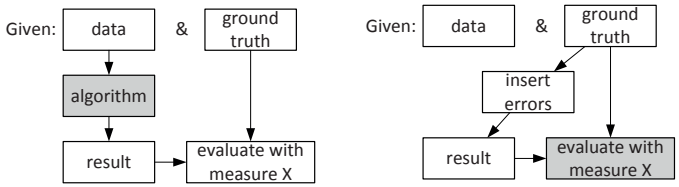


Figure 13.3: Traditional workflow to evaluate *algorithms* (left) & our workflow to evaluate *measures* (right)

Evaluation Setup We created a benchmark database with 5000 objects and 30 dimensions. Objects are grouped in 9 subspace clusters, each with 9 dimensions and covering 500 objects. The ground truth for this synthetic data is given. In traditional evaluation, to compare *different algorithms*, one uses each algorithm to obtain a clustering result and compares this result with the ground truth by using a specific measure (cf. Fig. 13.3 (left)). In our work, however, we want to compare the characteristics of *different measures*. Thus, instead of using an arbitrary algorithm on the data, we systematically insert specific errors into the ground truth to obtain an (imperfect) clustering result. This insertion of errors corresponds to an algorithm that is not able to detect the perfect clustering on the data. However, we are now able to analyze each type of error separately, leading to a more insightful analysis. Based on this clustering result, we can apply the different measures and evaluate their properties (cf. Fig. 13.3 (right)).

We created different clustering results based on our benchmark data, i.e., we insert different types of errors. Obviously, the perfect clustering result is simply the list of hidden subspace clusters ($Res = Ground$). This is the ideal baseline for all evaluation measures. However, we construct more realistic scenarios to analyze the sensitivity of the measures. For the first experiment, we split the hidden clusters into multiple found clusters regarding their objects, i.e., for each hidden cluster, we partition its objects into x parts of equal size, thus getting x smaller clusters. For grid-based subspace clustering approaches [AGGR98, NGC01, PJAM02, YM03], this might happen due to the discretization of the data space. Thus, we get a clustering result Res' that does not perfectly represent the ground truth. It is of major importance that evaluation measures are aware of such phenomena and show significantly lower quality if hidden clusters are split up. For the following experiments, we give short descriptions of the evaluation setup in the respective paragraph.

Object Awareness Most measures compare the set of hidden objects with the found objects. Though, there are major differences in their sensitivity to *splits of clusters*. As depicted in Fig. 13.4, the measures RNIA and F1-Merge do not detect these errors and provide constantly perfect values. The merge operation of F1-Merge is a clear drawback as a split of clusters does not affect the overall quality. A similar argument holds for RNIA because this measure only assesses the coverage of the hidden clusters but not whether the objects belong to the correct clusters. The Accuracy measure can still identify a reasonable generalization for the data with the split parts; thus, the quality decreases only slightly.

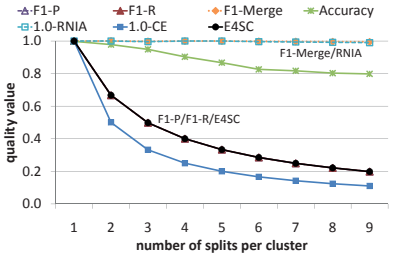


Figure 13.4: Split of clusters

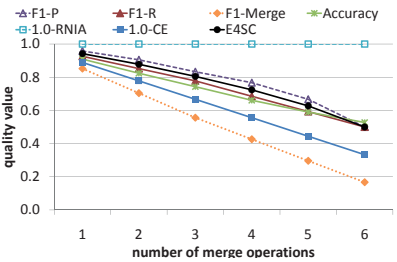


Figure 13.5: Merge of clusters

Next, we perform a certain number of *merge operations* between the hidden clusters. This is a realistic scenario where clustering algorithms fail to separate two similar clusters, resulting in a bad purity of clusters. As depicted in Fig. 13.5, almost all measures detect this bad clustering and show decreasing values with increasing amount of merges. Only the RNIA measure is not affected by merged clusters due to the reason described above.

Next, we consider object awareness for results produced by projected clustering methods [AWY+99, AY00b, PJAM02, YM03], which *partition the data*. However, as we have overlapping hidden clusters, a partitioning of the data cannot be the perfect clustering result. The resulting quality values range from 0.735 (RNIA, CE) to 0.847 (F1 measures, E4SC). Thus, all measures show lower quality values as each object is assigned to exactly one cluster and not multiple ones.

Overall, the first experiments have shown that most measures are aware of the most basic properties of clustering quality. Missing objects due to partitioning, affects the quality of all measures. Splitting or merging clusters affects all but the RNIA measure due to the simple coverage method.

Subspace Awareness In the following experiments, we consider errors in both, the detected object set and the dimension set, for each subspace cluster. We *remove/include objects and dimensions* respectively with a certain percentage. As depicted in Fig. 13.6, the perfect result is at the center of each figure (0% additionally included objects, 9 dimensions). For the x-axis, 'detected objects' $-p\%$ means that $p\%$ of the objects were excluded from the found clusters and $+p\%$ refers to the amount of included objects. For the y-axis, we added or removed dimensions compared to the perfect result respectively. We show four different measures (results of F1-R & F1-Merge are similar to the one of F1-P; the result of RNIA corresponds to the one of CE).

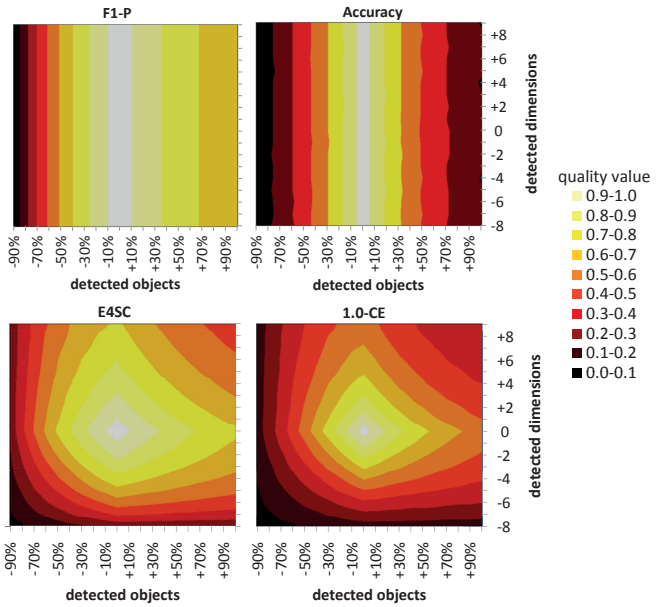


Figure 13.6: Removal and adding of clustered objects and relevant dimensions

Obviously, F1-P and Accuracy, as measures without subspace awareness, are only affected by changes in the object set; however, they react differently. While F1-P still returns good qualities when adding many objects, the Accuracy measure is more sensitive and exposes the area of perfect quality more distinctly. Our novel E4SC measure and CE are also affected by changes in the dimension set. Beside this subspace awareness of CE and E4SC, we observe interesting characteristics. While in the area of perfect clustering both changes in the object and dimension set affect the overall quality to the same degree (nearly rectangular areas), this effect is not observed for extreme cases where, e.g., most of the objects have been removed. For such extreme cases either the object set or the dimension set is dominating (rounded areas). Furthermore, the CE measure reacts more sensitive in this experiment, i.e., the quality values decrease faster than for E4SC. This property is especially observed in the first quadrant, where objects and dimensions are added.

Overall this experiment demonstrates the lack of subspace awareness in F1-P, F1-R, and F1-Merge. Their use in subspace clustering is at least questionable. E4SC and CE simultaneously handle subspaces and object groups.

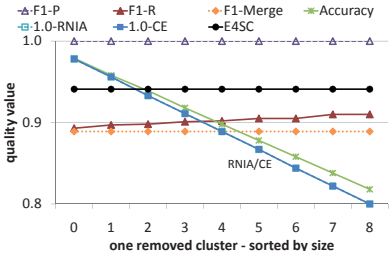


Figure 13.7: Miss of single clusters

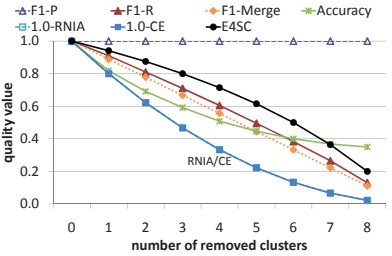


Figure 13.8: Varying miss of clusters

Set of detected clusters While in the previous experiments, we have analyzed different effects on the content of clusters, in the following, we consider typical clustering errors such as completely missing clusters or detecting similar clusters multiple times.

In our first experiment, depicted in Fig. 13.7, we *remove one cluster* from the resulting clustering each time. We generated 9 hidden clusters, each with a different number of objects. By sorting the clusters based on their object size, such that the smallest cluster is removed to the left of the x-axis and the largest one to the right, we show the *effect of the cluster size*. We observe that CE, RNIA, and Accuracy are affected more by the exclusion of large clusters than by exclusion of small ones. These measures are biased w.r.t. the cluster size. All other measures are nearly not affected by the size of clusters; they are unbiased w.r.t. the size. A special case is F1-P, that shows perfect quality although clusters have not been detected. As F1-P maps each found cluster to one hidden cluster, it is not affected by missing clusters (cf. Section 13.3).

Next, we exclude not only one cluster but accumulate the *removal of several clusters*, starting by excluding the largest cluster until only one cluster remains in the found clustering. As depicted in Fig. 13.8, all measures except for F1-P show decreasing quality values. However, the slopes vary greatly. CE, RNIA, and Accuracy show steep decrease in the first (large) clusters. Removing large clusters is more critical for these measures. F1-R and F1-Merge measures show almost linear decrease; removing one additional cluster results in constant drop of the quality. Our E4SC measure, however, decreases more quickly with each additionally excluded cluster. That is, excluding two clusters is more than twice as bad as the removal of a single cluster; although, we see a constant curve in Fig. 13.7.

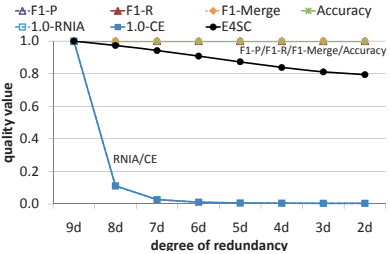


Figure 13.9: Increasing redundancy

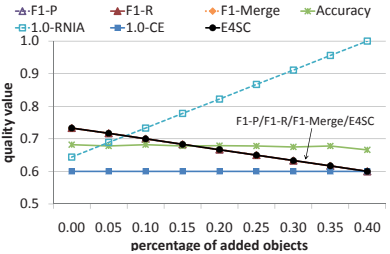


Figure 13.10: Impurity of clusters

Both, missing a cluster but also detecting similar clusters in redundant projections, should lead to lower quality values. In the following experiment, we additionally add *redundant clusters* in lower dimensional projections, as observed for several subspace clustering approaches [AGGR98, NGC01, KKK04]. The x-axis in Fig. 13.9 indicates the amount of redundant clusters. Starting by adding only the redundant 8-dimensional projections (of the original 9D clusters), we add more and more lower-dimensional ones. As illustrated, only CE, RNIA, and E4SC are affected by redundancy; quality decreases while redundancy increases. However, CE and RNIA show already a significant drop to almost zero quality for redundancy in 6- to 8-dimensional clusters; we get no discriminable quality values after this point. Our E4SC measure shows continuously decreasing quality, down to the 1-dimensional redundant clusters, but remains in high quality ranges.

Overall, these experiments have shown that F1-P, F1-R, and F1-Merge do not meaningfully assess the quality of redundancy polluted clusterings. Furthermore, F1-P is not able to detect a miss of clusters. For the criteria of redundancy and identification awareness, the measures CE, RNIA, and E4SC are preferable. However, for the presence of redundancy in the result, RNIA and CE easily assess a misleadingly low quality value, which the analyst needs to put into perspective.

Sensitivity of measures We create further evaluation scenarios to show the sensitivity of each measure. First, we create a clustering result where a certain fraction of the clustered objects is removed. This miss of objects is assessed as low quality by all measures in Fig. 13.10, as the leftmost values indicate. Afterwards, we progressively re-add objects to the clusters; however, not from the correct but different clusters. Thus, we increase the *impurity of the clustering results*. Most measures accurately show decreasing quality for impure clusterings in Fig. 13.10.

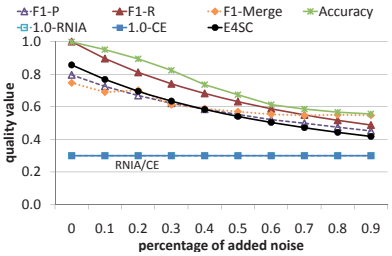


Figure 13.11: Noise polluted clusters

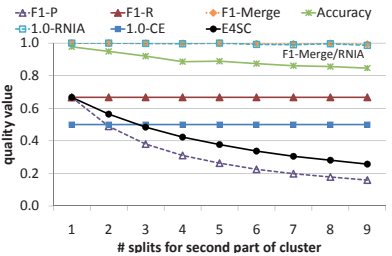


Figure 13.12: Split in unequal parts

CE and Accuracy, however, are almost not affected by this scenario. Because the largest part of the hidden cluster is still detected, adding incorrect objects does not influence their quality values. Even worse, the RNIA measure increases in quality since we cover more objects of the hidden clusters. Thus, these three measures are not sensitive enough to the impurity of clusterings.

Next, we simulate the effect of algorithms that are not able to detect *outliers*, i.e., algorithms enforcing an assignment of each object to a cluster. Thus, besides the hidden clusters, we generate one single cluster that contains all *noise objects* and add it to the clustering result. Most of the measures are influenced by this as the leftmost entries in Fig. 13.11 indicate. In the following, we relocate more and more objects from the noise-cluster to the hidden clusters. This phenomenon is typically observed for algorithms mixing up noise objects with the detected clusters. Clearly, the quality of the clustering should decrease because the true clusters are now polluted by noisy objects. However, CE and RNIA are not affected by noise as the hidden clusters are still covered and the purity is incidental. All other approaches show decreasing quality.

In the following experiment, we *split* each hidden cluster *not into equally large parts* but in one half of the cluster and an increasing fraction of further parts. While the large part seems to be the most valuable representation of the hidden cluster, the other parts hinder more and more the interpretation of the overall result set. As depicted in Fig. 13.12, most measures are not affected by this scenario because the major part of the hidden cluster is still found. F1-Merge and RNIA even show highest quality, ignoring the splits at all. Only F1-P and E4SC are aware of a decreasing quality.

Overall, for the core quality criteria, we have shown that measures as F1-P, F1-R, F1-Merge, and Accuracy are not aware of subspace properties and fail in

the very core of subspace cluster evaluation. Nevertheless, these measures are the most widely used ones in subspace clustering publications. They are of interest when only object sets are relevant or given for the evaluation. Enhanced measures such as CE, RNIA, and E4SC also consider the correct detection of subspaces. However, we have seen major differences in their evaluation. The RNIA measure is not affected by splits or merges of clusters. The CE measure shows high sensitivity to redundancy, while E4SC is sensitive to impure clusters, noise and splits of clusters. In Table 13.1, we provide an overview of the measures and their characteristics.

	F1-P	F1-R	F1-Mer.	Accu.	RNIA	CE	E4SC
object aware	+	+	-	-	-	-	+
subspace aware	-	-	-	-	+	+	+
redundancy aware	+	-	-	-	+	+	+
identification aware	-	+	+	+	+	+	+
split	+	+	-	+	-	+	+
merge	+	+	+	+	-	+	+
size unbiased	+	+	+	-	-	-	+
impurity	+	+	+	-	-	-	+
noise	+	+	+	+	-	-	+

Table 13.1: Characteristics of evaluation measures

Real world data and algorithms The previous experiments aimed at an objective analysis of all error types separately. Enabled by a systematic transformation of a presumed set of ground truth clusters for synthetic data, an unaffected consideration for each error type was possible. The following consideration of real world data and clustering results of real algorithms will now provide the opportunity not only to prove but also to apply the newly won insights to the measures. For these real world experiments, we used the publicly available clustering results of [MGAS09] and applied the discussed measures on them. Notice that the ground truth of real world data usually consists of a partitioning of the dataset and provides no subspace information with the partitions. Each table in Fig. 13.13, Fig. 13.14, and Fig. 13.15 shows the quality values of different subspace clustering algorithms for one dataset [MGAS09]. For each measure, we depict the maximal and minimal values obtained for various parameter settings [MGAS09]. To quickly grasp the main tendency of the results, we shaded the

quality values (gets darker with increasing quality). We only discuss Fig. 13.13 but similar observations can be drawn for the other datasets in Fig. 13.14 and 13.15. As a first observation, each measure yields different quality values for the same algorithms. Consequentially, each measure assesses a different algorithm as vanquisher. This is hard to understand if one is not aware of the underlying characteristics of evaluation measures. Thus, further interpretation requires information about the measures' sensitivity as presented in this work.

Diabetes (size: 768; dim: 8)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.70	0.39	0.72	0.69	0.03	0.01	0.14	0.01	0.4	0.09
DOC	0.71	0.71	0.72	0.69	0.31	0.26	0.92	0.79	0.39	0.12
MINECLUS	0.72	0.66	0.71	0.69	0.63	0.13	0.89	0.58	0.14	0.09
SCHISM	0.70	0.62	0.73	0.68	0.08	0.01	0.36	0.09	0.11	0.08
SUBCLU	0.74	0.45	0.71	0.68	0.01	0.01	0.01	0.01	0.46	0.13
FIRES	0.52	0.03	0.65	0.64	0.12	0.00	0.27	0.00	0.1	0.06
INSCY	0.65	0.39	0.70	0.65	0.37	0.11	0.45	0.42	0.64	0.04
PROCLUS	0.67	0.61	0.72	0.71	0.34	0.21	0.78	0.69	0.45	0.25
P3C	0.39	0.39	0.66	0.65	0.56	0.11	0.85	0.22	0.57	0.57
STATPC	0.73	0.59	0.70	0.65	0.06	0.00	0.63	0.17	0.4	0.01

Figure 13.13: Results for the measures on diabetes data

The shading of the results reveals the rare differentiation of Accuracy, CE, and F1 since results only differ slightly and do not allow contrasty comparisons of different approaches. This effect is mainly due to the insensitivity of the measures to a variety of errors or, in the case of CE, the excessive punishment of redundancy and its size-bias. Only RNIA and E4SC show discriminative results between the approaches.

The evaluation results in Fig. 13.13 point out the importance of redundancy awareness. On average, the measures F1 and Accuracy, which are insensitive to redundancy, assign higher quality values to the resulting clusterings than CE, RNIA and E4SC. The worth of redundancy awareness becomes even more evident as we see that mainly the partitioning approaches DOC, MINECLUS, PROCLUS, and P3C, whose results are not redundant by definition, have high values for CE, RNIA, and E4SC. Only with these measures the benefit of non-partitioning subspace clustering approaches that successfully try to avoid redundancy, like INSCY and STATPC, is traceable as they yield better quality values than other approaches without redundancy model.

Overall, we see that our awareness criteria and the thorough evaluation on synthetic data helps to interpret quality assessment on real world data based on

different measures. We also see that only measures fulfilling the majority of our 4 criteria are able to carry out significant quality assertions. The more error cases a measure is sensitive to (cf. Table 13.1), the higher is the quality assessment, and, thus, the better the contrast of low and high quality ratings.

13.5 Conclusion

A fair and comparable quality assessment based on objective evaluation measures is a key component for knowledge discovery in databases. In this chapter, we analyzed the evaluation measures for the research area of subspace clustering. We could show that novel criteria for a meaningful evaluation of subspace clustering algorithms are needed and introduced four major criteria each measure has to fulfill: object, subspace, redundancy, and identification awareness. Based on these categories, an analysis of existing measures identified their drawbacks for applicability. As a consequence, we presented a novel evaluation measure that fulfills our general quality criteria. In an empirical study, we highlighted the characteristics for each measure in typical clustering scenarios.

As a conclusion, we recommend to use CE and E4SC measures in future evaluations as both highlight main subspace clustering properties. In combination with other measures such as RNIA they also bring out the reasons for a bad clustering quality as depicted in Fig. 13.10, where we observe a significant increase in RNIA while CE remains constant and E4SC decreases. This divergence in clustering evaluation measures indicates an incorrect assignment of objects to the detected clusters. With such knowledge about the characteristics of measures, improvements of the clustering result and the algorithm itself can be fostered.

This evaluation lays the foundation for a fair evaluation of algorithms in the area of subspace clustering. Besides the analysis of measures, however, it is necessary to provide a set of benchmark data that is annotated with the hidden clustering structure, which is still a challenge for future studies.

Pendigits (size: 7494; dim: 16)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.30	0.17	0.96	0.86	0.06	0.01	0.20	0.06	0.2	0.2
DOC	0.52	0.52	0.54	0.54	0.18	0.18	0.35	0.35	0.23	0.13
MINECLUS	0.87	0.87	0.86	0.86	0.48	0.48	0.89	0.89	0.27	0.27
SCHISM	0.45	0.26	0.93	0.71	0.05	0.01	0.30	0.08	0.14	0.06
SUBCLU	--	--	--	--	--	--	--	--	--	--
FIRES	0.45	0.45	0.73	0.73	0.09	0.09	0.33	0.33	0.22	0.11
INSCY	0.65	0.48	0.78	0.68	0.07	0.07	0.30	0.28	0.17	0.02
PROCLUS	0.78	0.73	0.74	0.73	0.31	0.27	0.64	0.45	0.34	0.32
P3C	0.74	0.74	0.72	0.72	0.28	0.28	0.58	0.58	0.28	0.28
STATPC	0.91	0.32	0.92	0.10	0.09	0.00	0.67	0.11	0.64	0.01

Shape (size: 160; dim: 17)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.31	0.31	0.76	0.76	0.01	0.01	0.07	0.07	0.2	0.2
DOC	0.90	0.83	0.79	0.54	0.56	0.38	0.90	0.82	0.41	0.29
MINECLUS	0.94	0.86	0.79	0.60	0.58	0.46	1.00	1.00	0.42	0.3
SCHISM	0.51	0.30	0.74	0.49	0.10	0.00	0.26	0.01	0.26	0.15
SUBCLU	0.36	0.29	0.70	0.64	0.00	0.00	0.05	0.04	0.09	0.09
FIRES	0.36	0.36	0.51	0.44	0.20	0.13	0.25	0.20	0.28	0.21
INSCY	0.84	0.59	0.76	0.48	0.18	0.16	0.37	0.24	0.53	0.21
PROCLUS	0.84	0.81	0.72	0.71	0.25	0.18	0.61	0.37	0.31	0.2
P3C	0.51	0.51	0.61	0.61	0.14	0.14	0.17	0.17	0.2	0.2
STATPC	0.43	0.43	0.74	0.74	0.45	0.45	0.55	0.55	0.62	0.62

Vowel (size: 990; dim: 10)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.23	0.17	0.64	0.37	0.05	0.00	0.44	0.01	0.13	0.05
DOC	0.49	0.49	0.44	0.44	0.14	0.14	0.85	0.85	0.25	0.16
MINECLUS	0.48	0.43	0.37	0.37	0.09	0.04	0.62	0.34	0.12	0.12
SCHISM	0.37	0.23	0.62	0.52	0.05	0.01	0.43	0.11	0.11	0.09
SUBCLU	0.24	0.18	0.58	0.38	0.04	0.01	0.39	0.04	0.03	0.03
FIRES	0.16	0.14	0.13	0.11	0.02	0.02	0.14	0.13	0.03	0.03
INSCY	0.82	0.33	0.61	0.15	0.09	0.07	0.75	0.26	0.14	0.11
PROCLUS	0.49	0.49	0.44	0.44	0.11	0.11	0.53	0.53	0.13	0.13
P3C	0.08	0.05	0.17	0.16	0.12	0.08	0.69	0.43	0.25	0.21
STATPC	0.22	0.22	0.56	0.56	0.06	0.06	0.12	0.12	0.28	0.28

Figure 13.14: Comparison of evaluation measures for the datasets Pendigits, Shape, and Vowel of the UCI repository [FA10] based on the clustering results of different subspace clustering algorithms provided by [MGAS09]

Liver-Dis. (size: 345; dim: 6)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.68	0.65	0.67	0.58	0.08	0.02	0.38	0.03	0.2	0.05
DOC	0.67	0.64	0.68	0.58	0.11	0.07	0.51	0.35	0.09	0.05
MINECLUS	0.73	0.63	0.65	0.58	0.09	0.09	0.68	0.48	0.06	0.05
SCHISM	0.69	0.69	0.68	0.59	0.04	0.03	0.45	0.26	0.12	0.11
SUBCLU	0.68	0.68	0.64	0.58	0.11	0.02	0.68	0.05	0.06	0.06
FIRES	0.58	0.04	0.58	0.56	0.14	0.00	0.39	0.01	0.22	0.01
INSCY	0.66	0.66	0.62	0.61	0.03	0.03	0.42	0.39	0.06	0.03
PROCLUS	0.53	0.39	0.63	0.63	0.26	0.11	0.66	0.25	0.36	0.36
P3C	0.36	0.35	0.58	0.58	0.55	0.27	0.96	0.47	0.68	0.68
STATPC	0.69	0.57	0.65	0.58	0.23	0.01	0.58	0.37	0.32	0.02

Glass (size: 214; dim: 9)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.51	0.31	0.67	0.50	0.02	0.00	0.06	0.00	0.21	0.16
DOC	0.74	0.50	0.63	0.50	0.23	0.13	0.93	0.33	0.18	0.17
MINECLUS	0.76	0.40	0.52	0.50	0.24	0.19	0.78	0.45	0.28	0.15
SCHISM	0.46	0.39	0.63	0.47	0.11	0.04	0.33	0.20	0.15	0.14
SUBCLU	0.50	0.45	0.65	0.46	0.00	0.00	0.01	0.01	0.38	0.26
FIRES	0.30	0.30	0.49	0.49	0.21	0.21	0.45	0.45	0.24	0.24
INSCY	0.57	0.41	0.65	0.47	0.23	0.09	0.54	0.26	0.18	0.14
PROCLUS	0.60	0.56	0.60	0.57	0.13	0.05	0.51	0.17	0.21	0.07
P3C	0.28	0.23	0.47	0.39	0.14	0.13	0.30	0.27	0.29	0.28
STATPC	0.75	0.40	0.49	0.36	0.19	0.05	0.67	0.37	0.25	0.12

Breast (size: 198; dim: 33)

	F1		Accuracy		CE		RNIA		E4SC	
	max	min	max	min	max	min	max	min	max	min
CLIQUE	0.67	0.67	0.71	0.71	0.02	0.02	0.40	0.40	0.04	0.04
DOC	0.73	0.61	0.81	0.76	0.11	0.04	0.84	0.07	0.12	0.06
MINECLUS	0.78	0.69	0.78	0.76	0.19	0.18	1.00	1.00	0.12	0.08
SCHISM	0.67	0.67	0.75	0.69	0.01	0.01	0.36	0.34	0.01	0.01
SUBCLU	0.68	0.51	0.77	0.67	0.02	0.01	0.54	0.04	0.02	0.02
FIRES	0.49	0.03	0.76	0.76	0.03	0.00	0.05	0.00	0.02	0.01
INSCY	0.74	0.55	0.77	0.76	0.02	0.00	0.24	0.11	0.02	0.02
PROCLUS	0.57	0.52	0.80	0.74	0.51	0.11	0.65	0.43	0.48	0.16
P3C	0.63	0.63	0.77	0.77	0.04	0.04	0.19	0.19	0.03	0.03
STATPC	0.41	0.41	0.78	0.78	0.16	0.16	0.33	0.33	0.28	0.28

Figure 13.15: Comparison of evaluation measures for the datasets Liver, Glass, and Breast of the UCI repository [FA10] based on the clustering results of different subspace clustering algorithms provided by [MGAS09]

14

Subspace Search and Visualization for Alternative Clusterings

For subspace clustering methods, the process of identifying interesting subspaces is integrated in the process of clustering. Subspace search approaches decouple both processes, which provides more flexibility regarding the choice of the subsequent clustering approach. In this chapter, we propose a novel method for the visual analysis of high-dimensional data in which we employ an interestingness-guided subspace search algorithm to detect a candidate set of subspaces. Based on appropriately defined subspace similarity functions, we visualize the subspaces and provide navigation facilities to interactively explore large sets of subspaces. Our approach allows users to effectively compare and relate subspaces with respect to involved dimensions and clusters of objects. It supports the reasoning about the data from different perspectives, effectively yielding a more complete view on high-dimensional data.

14.1 Introduction

In this chapter, we propose a visual-analytics workflow for exploring high-dimensional data in subspace projections, making use of algorithmic subspace search in combination with visual-interactive representations for user-based filtering and exploration. The analysis of high-dimensional data does not only pose challenges to the cluster analysis but also to the visual-interactive side of data exploration. A limited number of available visual variables and limited short-term memory of human analysts make it difficult to effectively visualize data in high numbers of dimensions. As for clustering, restricting the visual analysis to only the most relevant feature subset will help to capture interesting data patterns more precisely. For the clustering analysis, we know that insights may not be hidden in only one single subspace, such that an analysis should consider multiple subspaces as well as their interrelations. Especially, for high-dimensional data, we can expect to have different views on the same data, whose ignorance might abandon useful information. The existence of alternative relevant subspaces may stem from the data description process when during preprocessing, features (dimensions) which describe different semantic properties of the data, are combined. For instance, in demographic analysis, households are often described by an array of many variables, combinations of which constitute different conceptual domains, such as wealth, mobility, or health. Likewise, it may be the combination of otherwise not semantically related dimensions, which by their combination give rise to interesting patterns. A class of *subspace analysis* algorithms has been proposed to cope with the problem of identifying interesting subspaces and clusters from a high-dimensional dataset. To date, however, there has been a very limited focus on the presentation and interpretation of the generated output. Furthermore, subspace analysis often produces highly redundant results that need to be further manipulated in order to get meaningful results [MAG+09a].

We propose an initial step towards the use of visual analytics as a way to explore alternative views generated by subspace analysis algorithms. We define an analytical pipeline made of algorithmic and visual components that permits to single out and explore alternative views in the data. After being analyzed by a subspace search algorithm, the data is structured and further processed in an interactive visualization environment to reduce redundancy. The main contribution of our approach is the operative definition and implementation of this multistep pipeline which permits to sift through an exponential number of sub-

space candidates and to reduce the problem to a handful of relevant views. More specifically, we (1) introduce a mechanism to deal with subspace redundancy by defining topological and dimensional subspace similarity and by allowing flexible and interactive subspace aggregation; (2) we provide a well-reasoned interactive visualization environment that permits to compare and assess alternative views by visually comparing topological and dimensional similarities and strike a balance between the visual complexity and the level of detail.

We evaluate our method through two case studies. The first one is based on synthetic data to demonstrate the tool's potential. The second one is based on real-world data to verify the tool's usefulness for finding and interpreting alternative views in high-dimensional data. We believe these results show the potential of visual analytics in the context of automated mining algorithms for supporting the understanding of the results, which can lead to new questions concerning more effective mining algorithms.

14.2 Subspace Analysis

In this section, we discuss the challenges for visual subspace analysis in more detail and explain how we tackle these with our new interactive, explorative framework supported by subspace search algorithms.

As is commonly known in subspace clustering, dealing with high-dimensional data in its subspace projections faces two main challenges. The first, serious challenge is a reasonable scalability w.r.t. the dimensionality of the dataset. As for a d-dimensional dataset the number of possible subspaces $S \subseteq \{1, \ldots, d\}$ is $\sum_{k=1}^{d} \binom{d}{k} = 2^d - 1$, many subspace clustering approaches do not scale well for very high-dimensional data. Every algorithm has to employ some strategy and heuristics to cope with such an exponential search space. The second, closely related challenge is dealing with high redundancy, that stems from the high similarity of the exponentially many subspaces. If two subspaces share a high proportion of dimensions, they are likely to exhibit a very similar clustering structure [GMFS09]. A large search result with high redundancy is, however, not beneficial for the user as it masks the complete information and is hard to interpret.

A core task in analysis of high-dimensional data is to apply a clustering method to reduce the data complexity and to identify groups of data for comparison. Different clustering algorithms follow different clustering notions, e.g., there exist density-based (e.g., DBSCAN [EKSX96]) or compactness-based (e.g.,

k-Means) clustering methods, and their outcomes often crucially depend on non-intuitive parameter settings. Usually several clustering attempts are required until the user has a useful result. It is obvious that high runtimes of subspace clustering processes (see Section 14.6.3) are not tolerable for such a workflow. Consequently, we decided to start the visual data exploration one step *before* the actual clustering process and decouple subspace search and the actual clustering. Dedicated subspace search algorithms have been designed to efficiently filter and rank the possible subspaces according to specific quality criteria (or interestingness measures). The different subspace search approaches usually depend on the mining task, such as clustering [BPK+04, CFZ99, KKKW03, NMB13] or outlier mining [KMB12, KMWB13]. Only [BKM+13] presents a more general framework, able to be adapted for both tasks. After subspace search has taken place, an arbitrary clustering approach can be used to cluster in the identified subspaces.

The use of subspace search for our purposes has several advantages: (1) It helps to effectively filter out those subspaces that, based on a low interestingness, do not need to be considered by the user. (2) Subspace search approaches are designed to reduce the search space efficiently and they do not need to compute clusters. And (3) although subspace search approaches themselves also rely on certain assumptions of what makes a subspace interesting for clustering, these assumptions do not necessarily lead to very different subspaces among different approaches. Therefore, the results are not as biased as they are for different clustering algorithms. For example, the quality assessment based on the k-NN distance [BPK+04] or based on the density within grid cells [CFZ99] both evaluate the proximity of objects within a subspace and probably will award similar subspaces. Furthermore they favor neither the DBSCAN nor the k-Means clustering notion. This enables the user to already obtain valuable results with a single subspace search approach. And (4) integrating the subspace search into the high-dimensional analysis offers the user the opportunity to obtain a visual, intuitive overview of the clustering structure *before* even starting the actual clustering. Thus, the user can assess the potential of the data to deliver valuable clustering results at all; select the subspaces to cluster in; decide which clustering notion to follow in each subspace (since the notion does not need to be the same for all); more easily determine meaningful parameter settings for clustering approaches.

Subspace search methods guide their search process by specific interestingness scores that are defined heuristically. For example, the method proposed in

[CFZ99] considers the variation of the density of objects across a regular cell-based partitioning of a given subspace. The underlying assumption is, that a higher variation of density provides a higher probability that the subspace shows a meaningful structure. As another example, the SURFING method [BPK+04] relies on the histogram of the k-nearest neighbor distances for all objects in a given subspace. It considers subspaces with non-uniform distance distributions more interesting, as they are an indication of the presence of a strong clustering. The underlying assumption is that for subspaces that show meaningful structures (e.g., clusters), different k-NN distances will occur. These and other measures aim at identifying subspaces that show a high "contrast" with respect to the distribution of objects, allowing to spot meaningful structures in the subspaces.

Subspace search methods also typically contain heuristic approaches to early abandon uninteresting subspaces, as exhaustive search would be prohibitively expensive. SURFING, for example, is based on a bottom-up strategy for searching subspaces by increasing dimensionality. It is based on testing additional dimensions for subspaces already known to be interesting. The list of currently interesting subspaces is continuously pruned to keep only the most interesting subspaces and speed up the search. SURFING has no dimensionality bias (i.e., does neither favor subspaces of lower dimensionality nor subspaces of higher dimensionality), assumes no specific clustering structure, and in practice, it is parameter free. Due to these properties, we rely on this method in our proposed approach, using the implementation provided to us by the original authors, but other subspace search algorithms could be easily used as well.

Overall, using the results of a subspace search algorithm as a starting point for our visualization has many advantages. Subspace search methods, such as SURFING, employ efficient search strategies tackling the efficiency challenge of subspace analysis. However, they typically do not solve the challenge of high redundancy, which motivates our proposed visual analytical workflow. Note that the recently introduced subspace search approaches [NMB13, BKM+13] take the redundancy explicitly into account algorithmically but have not been published at the time of this work.

14.3 Proposed Analytical Workflow

We propose a carefully designed visual-analytics workflow for subspace-based exploration of high-dimensional data, making use of algorithmic subspace search in

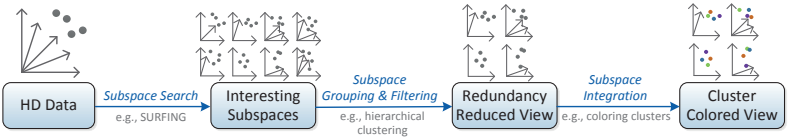


Figure 14.1: Our proposed analysis pipeline. A subspace selection algorithm is applied to automatically identify a candidate set of interesting subspaces. A filtering step reduces the potentially large and redundant set of automatically obtained subspaces to a user-selectable number of representing subspaces. Visual-interactive user exploration then proceeds on the subspace representations. Subspace analysis is also supported by comparative cluster views, allowing users to identify meaningful similar, complementary, or even conflicting clustering structures in the set of subspaces.

combination with visual-interactive representations for user-based filtering and exploration. Our approach starts (1) with an automatic *subspace search step*, where a large number of interesting subspaces is selected by a subspace search algorithm. Current subspace search methods provide an algorithmic handling of the problem of finding interesting subspaces, yet they often produce too many subspaces that may also be redundant and thereby overwhelm the interactive analysis (see also Section 14.2). We, therefore, employ a *similarity-based grouping of subspaces* (2) and perform the interactive exploration of interesting subspaces based on a few group representatives. Appropriate visual representations and interactions support the *visual interactive analysis* (3) for better understanding the subspace search results, including the support for comparative cluster analysis.

Figure 14.1 depicts our proposed analytical workflow. Next, we detail the technical design decisions made for each of the analysis steps, including discussion of alternatives.

14.3.1 Generation of interesting subspace candidates

The advantages for choosing subspace search, and in particular SURFING, have been already discussed in Section 14.2. We observe that typically subspace search algorithms output a huge number of subspaces. Since the examination of all subspaces is infeasible, a common approach is to filter the subspaces based on a certain threshold for the assigned interestingness value. This, however, ignores the fact, that the first ranked subspaces might be only slight variations (i.e., high overlap of dimension sets) of the same subspace and, therefore, are redundant

to each other. Yet, interesting subspaces with substantially different dimension sets, as compared to the top ranked results, could be found at much later ranking positions, and run the risk to be neglected from the analysis. Therefore, we apply a grouping step based on an appropriately defined notion of subspace similarity.

14.3.2 Similarity-based subspace grouping and filtering

Given a large number of candidate subspaces, we apply hierarchical grouping and filtering to yield a smaller set of mutually sufficiently different, yet individually interesting groups of subspaces for interactive analysis. Our filtering and grouping operation is based on a custom similarity function defined on pairs of subspaces according to two main criteria: (1) overlap of the sets of dimensions of the respective subspaces, and (2) resemblance in the data topology of the respective subspaces.

(1) Similarity based on dimension overlap: Subspaces can be similar regarding their constituent dimensions. We use the *Jaccard/Tanimoto Coefficient* [RT60] on bit vectors, indicating the contained (active) dimensions in a respective subspace (1 denotes an active dimension, 0 the converse). The Jaccard/Tanimoto Coefficient is then computed as the fraction of dimensions contained in both subspaces (AND-ing of the bit vectors), among the total number of different dimensions occurring in the subspaces (OR-ing of the bit vectors).

(2) Similarity based on data topology: We also compare subspaces with regard to their data distribution. Specifically, we consider the similarity of k-NN relationships in the respective subspaces. For efficiency reasons, we compute the k-nearest neighborhood ($k = 20$) lists for a sample of 5% of the contained data points. The similarity between two subspaces is then evaluated as the average percentage of agreement of k-NN lists in the subspaces. This score measures the similarity of the k-NN topology of the data, where k is a parameter and can be adapted to the datasets at hand by the user. Note that, in general, also other similarity measures are possible. For instance, the data can be clustered and the similarity between subspaces can be evaluated according to the resemblance of obtained clusterings by an appropriate measure such as the Rand Index [Ran71].

These two distance functions are the basis for the subspace grouping step in our analytical workflow:

(1) Subspace grouping: We apply hierarchical agglomerative grouping of subspaces based on the topologic distance function using Ward's minimum variance method [WJ63]. Based on the dendrogram representation of the obtained hierarchical grouping, the user chooses the hierarchy depth level to select a number of groups. This way the user can easily decide how many clusters to use for the analysis.

(2) Subspace filtering: Based on the previously achieved grouping of subspaces, we select one subspace from each group as representative: For each group we consider the subspaces with the lowest dimensionality and choose the one which exhibits the highest interestingness score. We note that other rules for filtering representatives are possible, but find this rule to be robust and effective for users, as it tries to keep the dimensionality as low as possible.

These steps together with both distance functions take us further towards our goal of understanding the different kinds of relationships between subspaces. They can complement, confirm, or contradict each other and the awareness of these relations can be crucial for further mining tasks.

		contained dimensions	
		similar	not similar
data topology	similar	truly redundant	confirmatory
	not similar	dominant dimensions	truly complementary

Table 14.1: Filtering cases that can be supported by our two defined subspace similarity functions.

Four basic cases can be identified, each of which might be relevant for a given subspace analysis task: (1) Subspaces that are similar in both, their dimension sets and their data topology (truly redundant subspaces); (2) Subspaces that are dissimilar in both, their contained dimensions and their data topology (truly complementary subspaces); (3) Subspaces that are similar w.r.t. data topology but dissimilar regarding their contained dimensions (confirmatory subspaces: we confirm the same data relationships in different subspaces); and (4) Subspaces that are similar w.r.t. their contained dimensions, but dissimilar regarding topology (this is generally not expected but could indicate the existence of one or a few dimensions which are by their nature very dominant for the data topology). Table 14.1 illustrates these four basic filtering cases.

14.3.3 Visual-interactive design

After hierarchical aggregation and/or filtering of the potentially redundant set of subspaces, we apply a set of analytical views for exploring and comparing the subspaces. Our displays are based on (1) scatterplot-oriented representations of individual subspaces or groups of subspaces, (2) similarity-based or linear list layouts for sets of subspaces, and (3) additional informative views, such as parallel coordinates and color-coding for the comparison of groups in the data.

The proposed design is the result of several iterations of alternative solutions in which we explored and compared several representations. Two design choices are worth discussing here: (1) the design of a visual representative for subspaces and (2) their layout. We decided to represent subspaces with scatter plots because they allow for the identification and comparison of groups in the data. More abstract representations like simple colored marks would require less space but would not allow the rich topological comparison provided by the scatter plots. In contrast, representations that are more complex like, e.g., parallel coordinates would provide a direct representation of the dimensions included in the subspace but would make their representation much more cluttered. As for the layout, we tried several tree and graph layouts to make the relationship between the subspaces and their shared dimensions explicit but we found that this rarely provides interesting insights and makes the visualization too cluttered to be of any use.

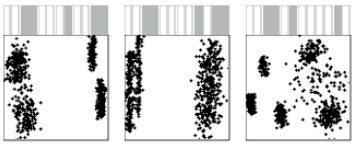


Figure 14.2: Subspace representation by 2D scatterplots with dimension glyph. We can see two 5D subspaces (left) and one 4D subspace (right) in the visual representations.

Scatter plots for subspaces can be generated by any appropriate projection technique, such as PCA [Jol02], MDS [CC94], or t-SNE [vdMH08]. We currently use MDS, but we experimented with others and any other technique could be used as an alternative. For a group of subspaces, one representative subspace is chosen (see below). To convey the involved dimensions, we also add an index glyph to the respective scatter plot (see Figure 14.2).

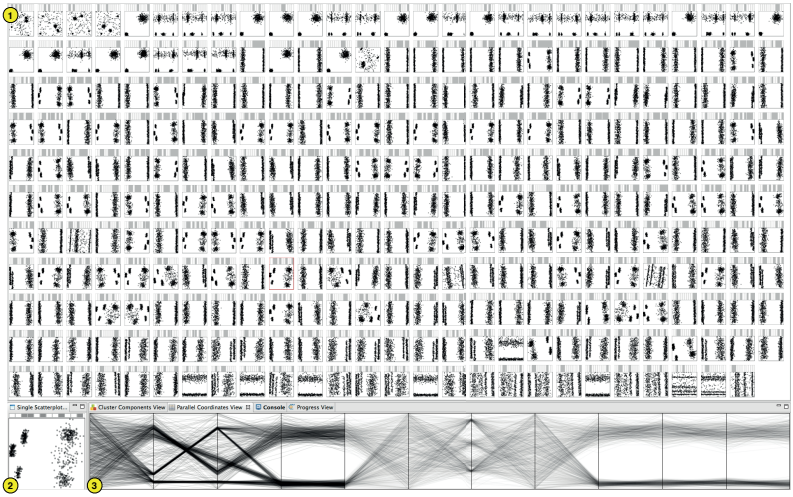


Figure 14.3: (1) *Linearly sorted view* of subspaces for the 12D synthetical dataset from [FBT+10] showing the full result of SURFING, consisting of 296 subspaces. The selected subspace in this view is shown in a (2) *single subspace view* to enable interaction and in (3) a *parallel coordinates view* with the subspace dimensions as the first axes (highlighted), and all the other data dimension as the last axes.

The analytical views are combined and linked in an application that consists of the following components:

Linearly sorted view of subspaces. To obtain a first overview of the output of the subspace search algorithm, we present all the subspaces in a linear view. The MDS scatter plots representing the individual subspaces are sorted left-to-right and top-down according to the interestingness index provided by the subspace search method. This view is exclusively used as a detail view for groups of topologically similar subspaces. Figure 14.3(1) illustrates the subspaces of the synthetic dataset, which is described also later in Subsection 14.4.1.

Subspace group view. In this view, groups of subspaces that have been formed by hierarchical agglomerative grouping are shown. Each group is represented by one selected subspace from that group, using the filtering method as described in the previous Subsection.

The representative subspaces are each visualized by an MDS plot, and shown side-by-side (Figure 14.4(1) illustrates). A dimension histogram on top of it indicates the distribution of dimensions contained by the subspaces in the group,

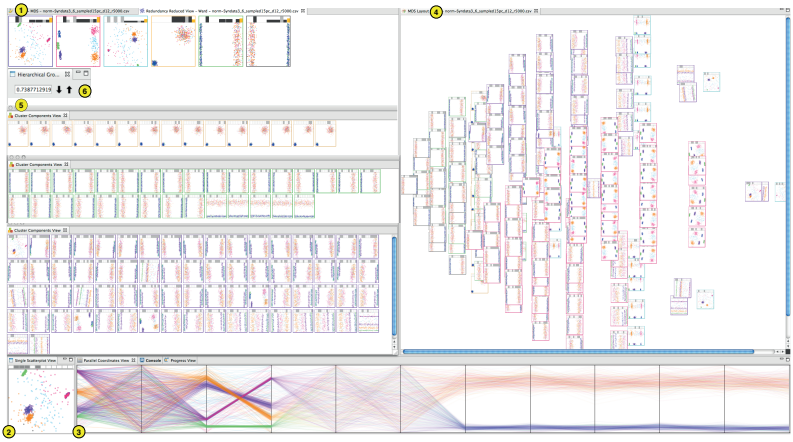


Figure 14.4: (1) *Subspace group view* for the $12D$ synthetic dataset with six subspace groups. (2) *Single subspace view* showing the representative subspace for the first group. (3) Details-on-demand in the *parallel coordinates view* for the selected subspace. (4) The MDS layout of the subspace search results based on their dimension similarity. (5) *Group detail view* for the three (orange, green, purple) subspace groups. (6) Hierarchical navigation buttons.

where the length of the bar encodes the frequency of the respective dimension. The last bar encodes the percentage of subspaces contained in this group. It is colored in orange to be easily distinguished from the others. Each group of subspaces from the preceding view can be expanded and its member subspaces can be seen and compared in detail (Figure 14.4(5)). This allows a better understanding of the current similarity threshold, and allows to expand or further collapse the group structure based on visually perceived similarity between subspaces. The user can investigate how similar the distribution of dimensions is among different groups of subspaces. To this end, a click on the dimension histogram icon of one particular group will cross-highlight the dimensions of the selected group that are also contained by other clusters. In summary, the subspace group view allows a global comparison of non-redundant subspaces and their similarities concerning the contained data topology.

Dimension-based subspace similarity view. We also support the comparative analysis of all subspaces based on their similarity regarding the set of active dimensions. To this end, a global MDS layout, based on the Tanimoto distances between the subspaces, as described in Section 14.3.2, is generated. Fig-

ure 14.4(4) illustrates the subspace similarity view. For a high number of subspaces, this view can only provide an impression of the similarity relationships but by zooming in, more details become visible. The subspace group view (based on the data topology distance) and the dimension-similarity view (based on the Tanimoto coefficient) are linked by color-coding (outer frame coloring). Thereby, we can compare the similarity of subspaces by their topological and dimension-overlap-based similarity.

Additional views and cluster comparison support. We also integrated a details-on-demand functionality for each subspace by a *parallel coordinates view* (Figures 14.3(3) and 14.4(3)). Highlighting the relevant dimensions helps to understand the difference of the subspaces in more detail. Furthermore, interactive exploration of the subspaces is enhanced by a *single subspace view*, providing an enlarged view of a selected subspace scatter plot (Figures 14.3(2) and 14.4(2)). This view also allows to manually select clusters of objects with a lasso tool. Cross-coloring the selected points among the other subspaces and within the parallel coordinates plot allows comparative exploration of grouping structures – a core problem in making effective use of alternative subspaces.

14.4 Application

We now demonstrate the analytical capabilities of our proposed approach. First, we use synthetic data as a proof of concept and exemplify the suggested workflow. We show how that relevant subspaces can conveniently be identified. Then, we describe an explorative setting in which interesting findings in alternative subspaces of a real world dataset are obtained.

14.4.1 Application Scenario 1: Synthetic Data

We used a 750 record sample of the first 12D synthetic dataset presented in [FBT+10] (dataset No. 2). This dataset consists of four 3D Gaussian clusters and two 6D Gaussian clusters. The remaining dimensions contain uniformly distributed random noise. The first step of our approach is to determine the interesting subspaces of the high-dimensional dataset by running automatic subspace search using SURFING (see Section 14.3). This subspace search returns a total of 296 subspaces identified as interesting, out of the 4095 possible subspaces. To get a first impression of these subspaces, we use the **linearly sorted view of**

subspaces shown in Fig. 14.3, relying on MDS representations of the data in the subspaces, and sorted by the interestingness score in decreasing order.

The view shows the diversity of subspaces identified during the automatic step. The first elements in the first row of the view are very similar in terms of the point distribution (showing mostly scattered and spherical point distributions). However, at later positions, we also see other varieties of point distributions, including parallel stripe patterns, and stripes mixed with spherical patterns. In a normal (non-visual) analysis case, relying just on the subspaces ranked top by the interestingness score, the analyst might extract redundant information and might miss some of these different characteristics of the subspaces.

The overview also confirms that the subspace search did return a lot of redundant subspaces, judging by the shape of the MDS projection representations. The next step is, therefore, to group the subspaces according to their similarity, allowing the user to abstract to a smaller number of relevant subspaces to compare them in detail. We used our similarity function based on the data topology, creating a hierarchal agglomerative clustering. The navigation buttons, as shown in Fig. 14.4(6), allow the user to move through each dendrogram level and to find the desired level of redundancy. Here, the dendrogram was cut at 0.73, very close to the root. As a result, six groups are found and visualized by their representatives. Fig. 14.4(1) shows that the number of subspaces can considerably be reduced in a meaningful way by the user. The number of groups can be varied, and the user can also investigate different levels in the dendrogram hierarchy. In this data we quickly found that six groups is the right level of detail for further investigation.

We investigate the components of each group of subspaces in more detail. Fig. 14.4(5) shows the group detail view of the orange, green, and purple subspace groups as framed in Fig. 14.4(1). Topologically similar subspaces are grouped together to give the analyst an overview of the existing groups and, if needed, to further compare individual group components.

On top of the scatterplots a dimension histogram indicates the distribution of dimensions for each group. The last bar of the histogram is marked in orange and represents the percentage of subspaces contained in this group. It is scaled logarithmically, so that this bar is also visible for groups with few elements. A click on the dimension histogram of one group representative highlights its dimensions in all the other representatives. In Fig. 14.4(1) the green group was clicked. To understand why the green- and gray-framed groups are split, we can

consult the additional view in Fig. 14.4(4). It shows an MDS layout of all interesting subspaces based on the dimension overlap (Tanimoto) similarity. In this view, closeness of two subspaces corresponds to dimension similarity. We see that the green- and gray-framed cluster groups are located on the far left side in the plot. This shows us that the subspaces are similar in terms of dimensions, but being in different groups, they must show different topological similarity according to our similarity measure. This can be explained as all the subspaces of the gray-framed group contain dimension $d12$, while none of the subspaces in the green-framed group contains this dimension. This is visible by the bars in the dimension histogram of the gray-framed group. As it is not highlighted, it is not contained in the marked green-framed group. This dimension is obviously responsible for a different data distribution.

We can also go one step further and compare the topological similarity of subspaces by cross-color-coding clusters of points in the MDS representation. Our lasso tool allows the user to manually mark clusters of points in the MDS subspace representation, which allows to cross-compare the groupings among different subspaces. For example, we manually marked six separate clusters of points in the pink-framed subspace group (group number two in Fig. 14.4(1)) and assigned distinct colors. By analyzing the distribution of colors among subspace group representatives, we see that other subspaces merge some of these clusters and spread others, highlighting the differing data distributions.

Summing up, we can see how our visual analytics workflow helps to deal with the extensive number of possibly interesting subspaces in a natural overview-first based visual analytics workflow. In a first step, the SURFING approach reduced the number of subspaces of the 12 dimensional dataset from 4095 to 296 interesting ones. Since this set of subspaces still showed a high redundancy, in our next step, we grouped them using our topological similarity measure. Based on the grouped subspaces, further investigations can take place to compare the relations and distributions among points of data within the subspaces.

14.4.2 Application Scenario 2: Exploration/discovery

We will now demonstrate the exploratory functionalities of our proposed approach based on a real dataset. We analyze the USDA Food Composition Dataset (http://www.ars.usda.gov/), a full collection of raw and processed foods characterized by their composition in terms of nutrients. The database contains more

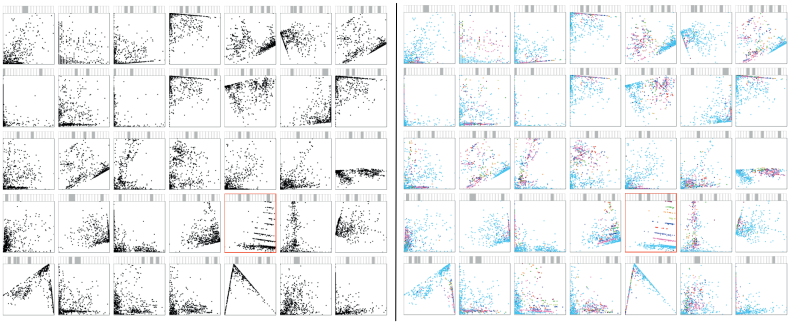


Figure 14.5: (1) *Linearly sorted view cutout* of subspaces for the 18D USDA Food Composition Dataset. The full result of SURFING, consisting of 216 subspaces. We see a rather high level of redundancy. Subspaces exhibiting more structure are found in particular at the mid and end positions in the ranking. Relying only on the numerically top ranked results, we would have omitted such interesting cases from the analysis.

than 7000 records and 44 dimensions. After removing missing values and outliers, as well as normalizing, 722 records (foods) remained for which we selected 18 dimensions of the dataset that where interpretable to us.

For this dataset, the application of the SURFING algorithm returned 216 interesting subspaces for further exploration. To get a first impression of this data, we investigated the *linearly sorted view* (see Fig. 14.5 for a cutout). Many subspaces, in particular those ranked with a high interestingness index, show a rather skewed distribution of points in our projection representation, concentrating along the edges of the diagrams. Only later in the ranking, we start to see the projections forming out more structure, that could be meaningful. The red color framed subspace in Fig. 14.5 seems to be very interesting, forming long, clear stripes. With the help of the *single subspace view*, we further investigated this subspace ($Iron, Maganase, Vit_D$) by coloring each stripe with a different color and compared the formation of these clusters across the other subspaces. Most of them seemed to be overspread by the cyan class (see Fig. 14.5 right).

At the same time, it is clear that a high level of redundancy is still present, and a further grouping is deemed necessary. Therefore, we continued with our next analytical step, the subspace grouping by agglomerative hierarchical clustering. We obtained different groups of subspaces and found out that these clearly striped clusters only appear in subspaces containing Vit_D.

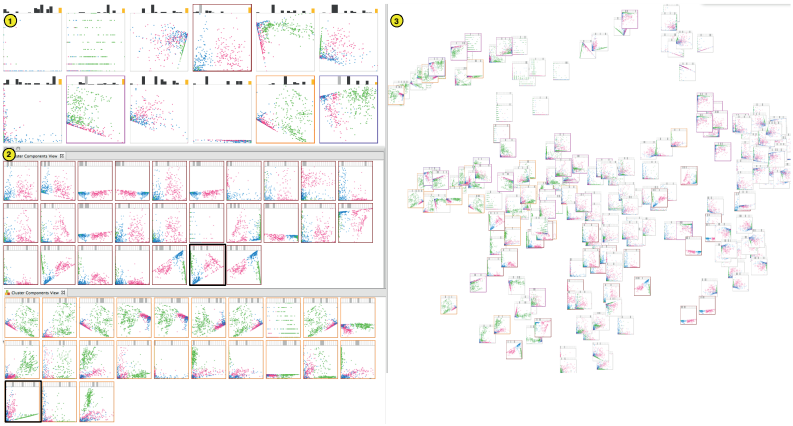


Figure 14.6: (1) *Grouped view* of subspaces for the 18D USDA Food Composition Dataset with 12 group representatives. (2) The brown and orange group components are shown in the *components view*. (3) MDS Layout of the total number of subspaces with cross-colored group representatives.

We therefore reset the coloring and started a new interactive analysis step, beginning with this stage of our workflow. After testing different filtering thresholds and comparing the topological- and the dimension-based similarity relations, we obtained a number of 12 groups, and considered this suitable for subsequent analysis.

From the reduced number of representative subspaces, one particular subspace stood out to us (see Fig. 14.6(1) for the group representatives and Fig. 14.7(A) for the interesting spotted one). This subspace shows the most structure and allows to discern two point clusters (pink and blue). We selected this specific subspace group (framed brown in Fig. 14.6) for further analysis. Cross-coloring is used to highlight its group components, that are shown at the bottom of the figure. It is visible that the group of subspaces are topologically similar, consequently this subspace is a valid representative.

In addition, we observe that there are some subspaces in this group where the clustering is changing. One example is shown in Fig. 14.7(B). We assigned the green color to the outstanding points on the left side, as they seem to form a different structure. In the group view (see Fig. 14.6(1)) we can see that this green cluster overspreads on five of the 12 subspace group representatives. After a closer look to the components of the orange subspace group, we spotted a

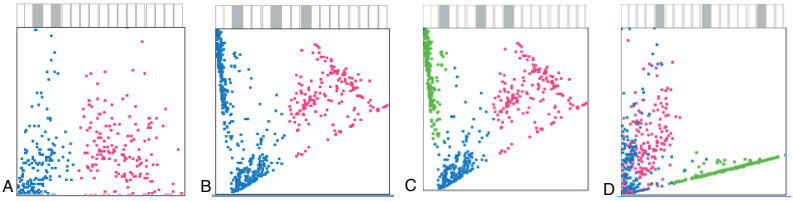


Figure 14.7: (A) Interesting spotted subspace ($Carbohydrate, Fibre$) presenting two clusters. (B) Subspace ($Carbohydarte, Lipid, Protein$) in the same cluster group of (A) where the cluster structure changes. (C) Green marked third cluster in subspace from (B). (D) Subspace ($Fiber, Protein, Vit_D$) of orange color-framed subspace group, where the alternative clustering of points is visible.

sharply defined green cluster (see Fig. 14.7(D) and highlighted in Fig. 14.6(2)). By highlighting the dimensions of the orange group, we can see that the brown group has a dominant dimension ($Protein$) that is not contained by any subspace of the orange group. We can therefore assume that this dimension is decisive for the clustering of the points. In the dimension-based similarity view (MDS Layout in Fig. 14.6(3)) the subspaces of the brown and orange groups are far apart from each other, which supports our finding that the groups contain different dimensions. Likewise we can see that the group components of the brown group are scattered across the MDS layout. This is due to the fact that the group subspaces are dissimilar in terms of their dimensions, but their topological similarity is dominated by the shared dimension ($Protein$).

Summing up, we demonstrated how our interactive, exploratory workflow can be applied to real data. Compared to the previous scenario, the information about the clusters is not known in real datasets, meaning that several interactive attempts are needed to investigate the vast number of interesting subspaces provided by the subspace search algorithm. With the help of the topological similarity functionalities, we could group the redundant clusters and have a closer look in their topological change. Using the different linked views of our approach helped us to identify different subspaces that present alternative clusterings.

14.5 Discussion and Possible Extensions

We will now summarize the main goal of our system, and discuss limitations and possible extensions.

14.5.1 Summarizing the Main Goals of our Approach

Our presented approach supports visual-interactive analysis of high-dimensional data from *multiple perspectives* based on the notion of automatic subspace search. The core assumption for our approach is that useful information could be extracted in a comparative way from several different subspaces residing in a larger high-dimensional data space. This assumption is the key driving force behind subspace search and subspace clustering algorithms developed in the Data Mining community over the past few years. We exploit algorithmic subspace search in an encompassing visual-interactive system. Our approach is designed around Shneiderman's Visual Information-Seeking Mantra [Shn96], applied to the problem of analyzing potentially large sets of subspaces. Modern subspace search methods such as SURFING efficiently identify candidate subspaces that are expected to exhibit informative structure without restriction to a specific nature of the structure. Specifically, interactively detecting and understanding relevant structures in subspaces is an explicit goal of our system. Our interactive support allows users to condense and compare subspaces, and even groups in data. Thereby, we close the analytical loop from algorithmic search of subspaces to sense-making by the user. Subspace search algorithms are very useful as a starting point. Since the identification based on interestingness is done heuristically, the search methods alone cannot solve the analytical problems at hand. To this end, capable visual-analytic systems need to be designed based on the output of the subspace search algorithm. We therefore designed, implemented, and applied an encompassing system design based on a subspace search method (exemplarily we used SURFING). It allows to explore high-dimensional data taking into account the curse of dimensionality and the possibility to find alternative clusters in different subspaces.

14.5.2 Limitations and Possible Extensions

We identify the following opportunities to improve our approach.

Computational scalability. We designed and tested our system around datasets of moderate dimensionality of tens of dimensions. For higher-dimensional data, we will have to deal with scalability issues regarding the computational complexity of the subspace search. To tackle this problem, subspace search algorithms probably need more aggressive filtering mechanisms to keep the number of searched subspaces in the exponential search space tractable. However, we

still need to ensure that no relevant results are excluded. The approaches of [NMB13, BKM+13] provide possible solutions to this problem.

Visual scalability. For higher-dimensional data, we also need scalable visual representations. We need to scale with the number of subspaces and the representation of each subspace. The linearly sorted view, per se, does not scale for many subspaces, yet it can be restricted to the representative subspaces obtained from the hierarchical grouping. A visual representation of subspaces is realized by lower-dimensional projections to show the data points and an index view to show the relevant dimensions. In particular, the latter will only scale for a limited number of dimensions. The design of set-oriented views to compare many sets of dimensions is a challenging problem whose solution would improve our tool.

Projection-based subspace representation. We currently represent the subspaces by MDS projections of the data residing in respective subspaces. However, projection typically induces loss in information, that could be incorporated in our visualization, e.g., by showing the stress values in an overlay visualization [SvLB10]. In our experiments, MDS performed very well compared to using PCA. Yet, it would be interesting to test other projections. Also, other subspace representations besides scatterplots could be thought of, in essence similar to Value-and-Relation displays [YPH+04]. Likewise, many different, useful similarity notions to group and compare subspaces, such as notions based on stress measures, implicit clustering structures, relations to outliers, Scagnostics features [WAG05], etc. could be employed. Testing them in different application domains is considered valuable future work. We note that our analytical approach can easily accommodate alternative subspace search algorithms, representations, and filtering options.

Interpretable Dimensions. To relate subspaces and data groups in subspaces, it is important for the analyst to be aware of the meaning of the dimensions of the respective subspace. Our index-based glyph does not convey information about the type of dimension. Detail-on-demand functions could be added to help the user interpret the involved dimensions and properties of the data points semantically and efficiently.

Definition of interestingness and sensitivity to noise. Subspace search algorithms heuristically identify subspaces as interesting based on certain properties of object relations. Based on the user and application, additional interestingness formulations are possible and should be supported. Following best practices in data analysis, we have applied a data cleaning step (outlier and missing value

removal) to our tested data before we fed it into our system. The SURFING algorithm is not robust with respect to missing values, whereas it seems to be robust with respect to outliers. The original paper does not discuss this aspect and we did not further investigate it. The projections used to represent data distributions in subspaces are sensitive to outliers and may generate clamped distributions if not pre-processed. We postpone the analysis of this problem to future work.

Usability and user adoption. Our current system design targets users with expertise in data mining. End-user applications, e.g., in market segment analysis, could benefit from subspace analysis. Though, we recognize that for end-users, the interface of our system would need to be customized, possibly. Our experience in collaborating with data mining experts showed that the tool can be useful not only for data exploration but also as an evaluation tool to assess the output generated by subspace analysis algorithms.

14.6 Related Work

14.6.1 Visualization and Clustering of High-Dimensional Data

Visualization of high-dimensional data is a long-standing research topic. Classic approaches include parallel coordinates, scatter plot matrices, glyph-based and pixel-oriented techniques [WGK10]. By appropriately sorting dimensions and mapping them to visual variables, these methods allow to overview and relate high-dimensional input data. However, we may experience scalability problems for large numbers of dimensions or records. Dimensionality reduction methods, such as PCA [Jol02] or MDS [CC94], can be used for the subsequent visualization.

Identification and relation of groups of data is a key explorative data analysis task. Often, user interaction is needed to identify and revise the number and characteristics of data clusters found by automatic search methods. To this end, visual-interactive approaches are useful. Although, many methods have been proposed, we can only highlight few of them in an exemplary manner. In [Shn02], interactive exploration of hierarchically clustered data along a dendrogram data structure is proposed to help users find the right level of clusters for their tasks. In [YWRH03], the parallel coordinates approach serves as a basic display to show data clustering results allowing to compare clusters in their high-dimensional data space. Also, 2D projections, possibly in conjunction with glyph-based representation of clusters, are widely employed, a recent example is [CGSQ11].

These approaches to visualization and clustering in high-dimensional data spaces all have in common that they are based on a given full (or reduced) dimensionality of the input dataset. Thereby, they show only a *singular* perspective of the usually multi-faceted high-dimensional data, that might not be the most relevant one. As we show with our approach, it is also useful to explore high-dimensional data for patterns in subsets of its full high-dimensional input space to increase potential data insight.

14.6.2 Automatic and Visual-Interactive Feature Selection

In machine learning, feature selection is the problem of selecting a small number of features from a larger input feature space such that a measurable criterion, e.g., the accuracy of a classifier [LM07], is optimized. Most automatic feature selection methods rely on supervised information (e.g., labeled data) to perform the selection. Therefore, they are not directly applicable to the explorative analysis problem. In existing works, involving visual-interactive selections or comparison of features, the Rank-by-Feature Framework [SS04] provides a sorted visual overview of the correlation among pairs of features. In [JJ09], the selection of input features was supported by a measure of the interestingness of the visual view provided by candidate features. An interactive dimensionality reduction workflow was presented in [IMI+10], relying on visual approaches to guide users in selecting features.

In [BvLBS11] and [BvLH+11], interactive visual comparison was proposed to relate data described in different given feature spaces based on 2D mappings and tree structures extracted from the different data spaces. Furthermore, in [LSP+10] a visual design based on network and heat map visualization was proposed to relate clusterings in different subsets of dimensions. In [YWRH03], dimensions are hierarchically clustered based on a simple value-oriented similarity measure. Based on this structure, user navigation can take place to identify interesting subspaces. In a recent work [YWG11], the output of this simple search method was visualized by tree- and matrix-based views, where each dimension combination was represented by a single MDS plot.

In summary, many of these methods are applicable to compare data regarding different criteria. However, most of them assume the feature selection to be performed globally and do not take the subspace search problem directly into account.

14.6.3 Subspace Cluster Analysis and Visualization

The problem of finding clusters in high-dimensional data can be divided into two sub-problems: *subspace search* and *cluster search*. The first one aims at finding the subspaces where clusters exist, the second one at finding the actual clusters. The large majority of existing algorithms considers the two problems simultaneously. Only few works consider a visualization support for subspace clustering. The VISA [AKMS07b] system uses visualization to help interpreting the subspace clustering result. A global view shows the similarity between clusters in terms of the number of records and dimensions, and a detail view shows properties of individual clusters. A disadvantage of this approach is that no visualization or comparison for the data distribution in the respective subspaces is supported. Heidi Matrix [VK09] uses a complex arrangement of subspaces on a matrix representation based on the computation of the kNN in each subspace. The complex visual mapping scheme may not be easy to use and its effectiveness, to the best of our knowledge, has not been evaluated yet. [FBT+10] proposes an approach for finding and visualizing interesting subspaces in astronomical data. Candidate subspaces are found from the data and ranked by a quality metric based on density estimation and morphological operators.

We note that if we apply one of these subspace clustering visualizations, we immediately inherit two main challenges of this paradigm that are still considered open research issues, namely: the efficiency challenge (relating to subspace cluster search) and the redundancy challenge (relating to the typical redundancy of the outputs generated).

14.7 Conclusions

We presented an encompassing visual-interactive system for subspace-based analysis in high-dimensional data. Subspace-based analysis can constitute a new paradigm for high-dimensional data analysis since informative structures in the data can be found and compared in different subspaces of a larger high-dimensional input space. We defined, implemented, and demonstrated an analytical workflow based on automatic subspace search. A large set of automatically identified interesting subspaces is grouped for interactive exploration by the user. A custom subspace similarity function allows for comparing subspaces. Our approach is able to effectively pin down several interesting views and helps to come

up with specific findings regarding similarities of groups in the data. We discussed a set of possible extensions of the system, which could be addressed as future work.

15

Interactive Analysis of Multiple Views

THE research area of subspace clustering is highly related to the research area of detecting multiple clusterings. Multifaceted data and its clustering results can often be semantically structured by concepts. Although subspace clustering methods generate concept-based patterns, the user has to provide domain knowledge to gain reasonable concepts out of the data. The first tool *CoDA* (*Concept Determination and Analysis*) that we present in this chapter supports the user in the final step of concept definition. More concretely, the user is guided through an iterative, interactive process in which concepts are suggested, analyzed, and potentially refined. The core aspect of *CoDA* is an intuitive, concept-driven presentation of subspace clusters such that concepts can be visually captured.

Based on the concepts defined with *CoDA* or for the several alternative clustering solutions generated by multi-view approaches, another interesting task is to learn how the different solutions are related to each other. Our second tool, *MCExplorer* (*Multiple Concepts Explorer*), allows for an interactive exploration, browsing, and visualization of multiple clustering solutions on several granularities. *MCExplorer* is applicable to the output of both full-space and subspace clustering approaches.

15.1 Introduction

In the past decades, a multitude of clustering methods were developed to extract one ('the best') grouping out of the data. However, data is often multi-faceted: for a single dataset, multiple valid interpretations and thus different alternative groupings are possible. Each of these valid views, also called *concepts*, provides different insights about the data. Accordingly, extracting these multiple or alternative clusterings is an active research area as recent publications in the field of full-space clustering show [CFD07, DQ08, QD09, BB06, GH05].

In this thesis, we focused on combining the research area of multi-view clustering with the research field of subspace clustering, where the observation of multiple valid groupings within the data is even more apparent: many subspace clustering algorithms implicitly detect multiple concepts by their very nature. Often, a single subspace clustering result corresponds to multiple views and certain clusters are a manifestation of an abstract concept. In the toy example of Chapter 4 (cf. Fig. 15.1) the subset of clusters "healthy living individuals" and "unhealthy living individuals" represent the concept "health status", while the set of clusters $\{C_3, C_4, C_5\}$ describes the concept of "taste of music".

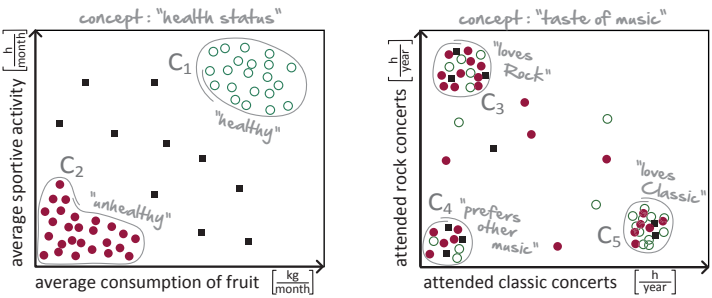


Figure 15.1: A subspace clustering revealing two different clustering concepts

While traditional subspace clustering techniques [PHL04, KKZ09] simply detect clusters in arbitrary subsets of the dimensions and concepts are only detected implicitly, some approaches (e.g., OSCLU and ASCLU (Chapters 4 & 5)) explicitly utilize the idea of concepts to identify clusters in strongly differing subspaces. The set of clusters generated by subspace clustering approaches is, however, unstructured and the different underlying concepts have to be determined in a post-processing step. The first tool CoDA, that we will introduce in Section 15.2,

helps to determine and analyze concepts for the output of any subspace clustering method and to assign the detected subspace clusters to their corresponding concepts. Thereby, CoDA helps in closing the gap between subspace clustering and multi-view clustering.

Extracting different concepts and their clusters is a first valuable step towards knowledge extraction. The similarly essential and indispensable second step deals with the analysis of detected patterns. Our tool MCExplorer, presented in Section 15.3, focuses on this second step and supports the comparison and analysis of the generated alternative solutions. This tool enables the user to decide on her preferences regarding different solutions but also to understand the different grouping patterns.

15.2 A Tool for Concept Determination and Analysis

15.2.1 Introduction

This thesis thoroughly discussed the fundamental parallels between subspace clustering and multi-view clustering. Although we can interpret the set of detected subspace clusters by one of the various subspace clustering approaches as manifestations of several abstract concepts, this set is unstructured and the concepts do not become apparent. Out of the various techniques for subspace clustering [KKZ09], some approaches as OSCLU (Chapter 4), ASCLU (Chapter 5), or [CFD07, DB13b] already focus on the specific task of grouping objects according to underlying concept structures: they find clusters in strongly differing subspace projections, providing the key for discovering the inherent concept structure. Since the concepts are generative, i.e., they actually induce the clusters, they cannot be automatically concluded out of clusters. Accordingly, the mentioned subspace clustering techniques achieve concept-based aggregations of objects but are not capable of abstracting from these aggregations in the sense of named concepts.

In real-world applications, however, the interest lies in the explicit discovery and naming of the underlying concepts. This task cannot be solved automatically by unsupervised learning methods such as subspace clustering but requires the domain knowledge of an expert. Our tool *CoDA* supports the user in revealing the

concepts out of a given subspace clustering. For this purpose, it provides the user with concept-oriented cluster visualization and interactive exploration to enable her to uncover the inherent concept structures. The main challenges tackled with *CoDA* are:

- Determination of concept structures
- Analysis of single concepts, as well as inter-concept dependencies

Each concept can be described by its occurring clusters on the one hand and its characteristic attributes on the other hand. Since the related clusters are not known beforehand, the idea is to capture the concepts through the structure of relevant attributes of the clustering. The relevant attributes are of particular importance for a semantic labeling of clusters and concepts. The process of concept determination can be divided into two phases as depicted in Fig. 15.2. Given a subspace clustering of database objects, in a first determination step, an interim grouping of clusters representing concepts is calculated based on their relevant subspaces. In a second determination step, the user sets the significant attributes for each represented concept. In the analysis phase, the user takes a closer look at the concept compositions and gives feedback to refine or to recalculate the concept structures. Thus, the whole process of concept discovery is iterative and highly dependent on user interaction.

In the following two subsections, we introduce our tool *CoDA* (*Concept Determination and Analysis*). *CoDA* is integrated into the OpenSubspace framework [MAG+09a, MGAS09] that adds subspace clustering functionality to the well-known WEKA Data Mining Software. In this framework, several subspace clustering algorithms are integrated; for *CoDA*, these algorithms can provide subspace clusterings to analyze them for concept structures. Fig. 15.3 shows a screenshot of the framework with the *CoDA* integration. Subspace clusterings are created in the *SubspaceClusterer* tab, and *CoDA* is realized in the *CoDA* tab. Since *CoDA* comprises two phases, i.e., concept determination and concept analysis, these phases are realized by two distinct tabs. The final concepts are determined by a cyclic usage of the two interdependent phases.

15.2.2 Concept Determination

In the following, we present how concepts are determined with *CoDA*. Remember, concepts induce clusters and not vice versa. Since in most application scenarios, the inherent concepts of the data are unknown, *CoDA* determines these concepts

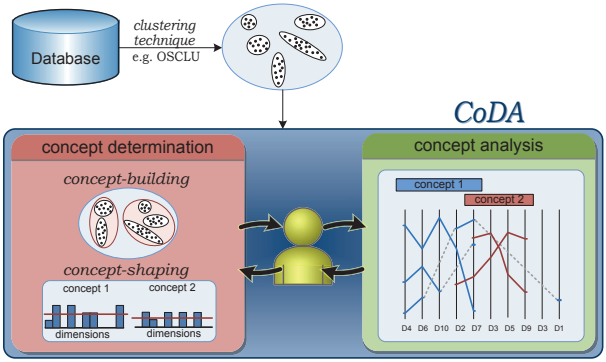


Figure 15.2: 2-phase, iterative workflow of *CoDA*

for a given subspace clustering by integrating users and their domain knowledge into the search process. The phase of concept determination has two goals: First, to assign the given clusters to possible concepts. Second, to determine the significant dimensions of these concepts.

Clusters that share relevant dimensions are expected to describe the same concept and are, therefore, automatically grouped together. These groupings, however, do not consider semantic knowledge; the user has to refine them in the concept analysis phase. The assigned clusters of a possible concept can have different relevant dimensions, preventing an automatic determination of the concept's significant dimensions. It is, therefore, the task of the user to select these dimensions. This process is called concept shaping. The two steps, cluster grouping and concept shaping, are now presented in more detail.

Finding concepts based on cluster grouping.
The first step aims at grouping subspace clusters such that the resulting groups possibly represent meaningful concepts. This is achieved by grouping the given subspace clusters according to their relevant dimensions and knowledge that was obtained in previous iterations of the concept analysis; the latter will be explained in more detail in Section 15.2.3. The clusters of one group very likely belong to the same concept and, therefore, represent this concept. In *CoDA*, the found concepts are displayed in the left part of the *concept determination* tab (cf. Fig. 15.3). The details of a concept's corresponding subspace clusters can be inspected by the user: by clicking on a cluster the cluster's objects and the relevant dimensions are shown. This is a functionality that is already implemented in the OpenSubspace framework and has shown to be very intuitive.

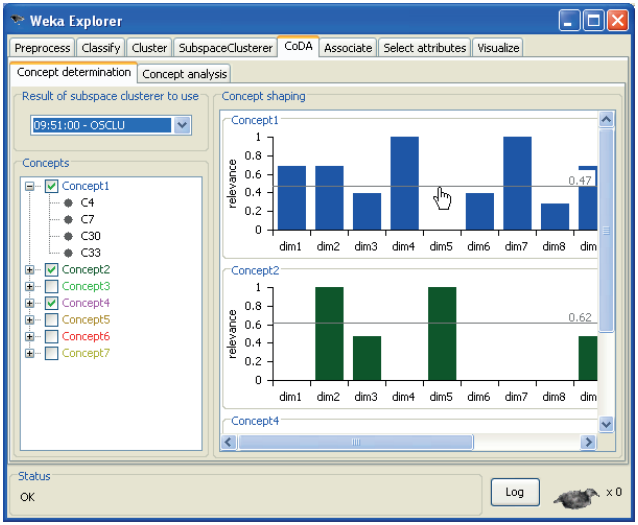
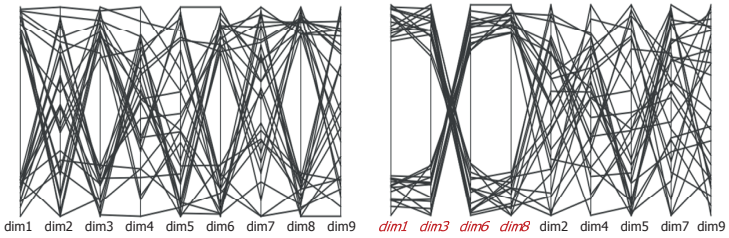


Figure 15.3: Concept determination tab of *CoDA*

Technically, the grouping of clusters to achieve meaningful concepts is realized by constraint-based clustering [WC00, WCRS01, Dav09]. In this clustering, the similarity between two clusters C_i and C_j is determined solely through their relevant dimensions, i.e., the similarity of their subspaces S_i and S_j. It is formally defined by (Jaccard/Tanimoto Coefficient): $sim(C_i, C_j) = |S_i \cap S_j|/|S_i \cup S_j|$. Knowledge obtained in previous iterations of the concept analysis is included into the clustering process by encoding this knowledge as constraints. More concretely, we provide must-links and cannot-links, i.e., the user can specify which clusters belong to the same concept and which do not.

Concept shaping.

In the following, we describe how the preliminary concepts found in the previous step are concretized by determining the significant dimensions of a concept. *CoDA* provides a bar chart for each concept that visualizes the relevance of each dimension (cf. Fig. 15.3). These dimensions are the basis for specifying the semantics of the final concepts in the preceding concept analysis phase. Since the corresponding subspace clusters of a concept have different relevant dimensions, the significant dimensions cannot be determined automatically. Based on a visual discrimination of significant and non-significant dimensions, the user can specify a threshold for each concept. Formally, the relevance of a single dimension d_i



(a) Naive dimension ordering (b) Concept-based ordering

Figure 15.4: Dimension ordering in parallel coordinates plots and their influence on visual interpretation

for a concept cpt and its assigned clusters $C_j = (O_j, S_j)$ with object set O_j and subspace set S_j, is determined by:

$$rel(d_i, cpt) = \frac{1}{\sum_{C_j \in cpt} |O_j|} \sum_{C_j \in cpt} |O_j| \cdot |\{d_i\} \cap S_j|$$

The output of this phase is a set of concepts and their selected significant dimensions.

15.2.3 Concept Analysis

In the previous phase of *CoDA*, the user determines the concepts; the second phase, described in the following, allows an in-depth analysis of these results (cf. Fig. 15.5). First, the analysis enables the user to comprehend the domain-specific semantic of a concept, e.g., by examining the actual characteristics of the clusters induced by the concept. Second, the user can improve the concept determination of subsequent steps by identifying any discrepancies in the current step.

Concept-centric parallel coordinates.
Our *CoDA* uses parallel coordinates to visualize the concepts and their induced subspace clusters in an intuitive way. Parallel coordinates are a technique to illustrate high-dimensional datasets [ID90]. Because each concept is associated only with a subset of dimensions, i.e., its significant ones, an intuitive illustration is challenging. A naive use of parallel coordinates would lead to a representation where significant and non-significant dimensions are interweaved. The example in Fig. 15.4(a) shows a plot of two subspace clusters of the same concepts and with the relevant dimensions $\{1, 3, 6, 8\}$. A visual interpretation of this plot

and, thus, a knowledge extraction is difficult since the non-significant dimensions hinder a condensed view of the data. For a clear representation it is important to group the significant dimensions of a concept together. In Fig. 15.4(b) the dimensions are permuted such that $\{1, 3, 6, 8\}$ are adjacent.

With our *CoDA* the user is able to analyze inter-concept dependencies, i.e., several concepts (with different sets of significant dimensions) are visualized simultaneously. To facilitate a clear visual impression for the user, *CoDA* performs a sophisticated arrangement of the dimensions, such that for each considered concept its significant dimensions are grouped together as good as possible. The arrangement of dimensions is easy to realize for each concept individually but when several concepts are considered simultanously the problem of arranging dimensions gets more complicated. Technically, the optimal ordering of dimensions is solved by using matrix bandwidth minimization techniques [MS00, MCP08].

The concept analysis tab of *CoDA* is depicted in Fig. 15.5. The user is able to select a set of concepts to be analyzed with our tool. Based on the selected concepts and their significant dimensions, the arrangement of the dimensions is determined automatically. The clusters of the corresponding concepts are plotted within the parallel coordinates diagram. By using color codes, the different concepts and their induced clusters can be distinguished. To enable a visual interpretation, *CoDA* describes a cluster by a single representation (e.g., the cluster mean) instead of all its objects, and we also skip the irrelevant dimensions of each subspace cluster. However, keep in mind that the relevant dimensions of the clusters do not necessarily correspond to the significant dimensions of the concepts. To provide the user with a comparison of these dimension sets and to give an easy overview for analyzing the inter-concept dependencies, *CoDA* additionally shows the significant dimensions of each concept in a bar diagram below the parallel coordinates.

User interaction for concept improvement.
By comparing the relevant dimensions of clusters and the significant dimensions of concepts, the user is able to detect any discrepancies in the concept determination so far. In Fig. 15.5, for example, the cluster $C4$ fits not very well to its currently assigned concepts. Adjusting the concept determination based on the concept analysis is thus crucial for a meaningful overall interpretation.

The easiest way to modify the current concepts is by readjusting the significance thresholds in the concept determination tab (cf. Fig. 15.3). Thereby, the user changes the significant dimensions of the concepts and consistency with the

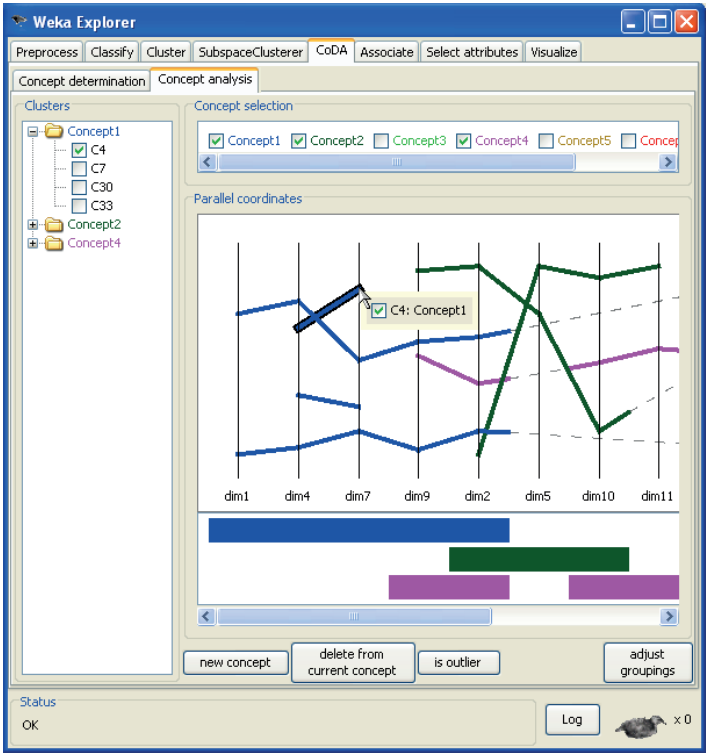


Figure 15.5: Concept analysis tab of *CoDA*

induced clusters can be realized. Note that this interaction does not influence the cluster grouping. For adjusting these groupings, *CoDA* implements more complex interactions such that the user is able to initiate a regrouping of the clusters to form novel concepts.

Consider the cluster $C4$ and the other cluster in the same subspace in Fig. 15.5. Based on the previous analysis and with the knowledge of the application domain, the user identifies that these two clusters do not belong to the current concept but they build an own concept. In *CoDA*, these clusters can be selected and the user can enforce this set to represent a new concept (button 'new concept' in Fig. 15.5). Similarly, the user can resolve conflicts if a cluster is wrongly assigned to a concept (button 'delete from current concept'): the selected cluster

has to be assigned to another concept. Even stricter, the user can classify clusters as outliers that do not belong to any concept (button 'is outlier'). The user's decision, which interaction is reasonable, can be further confirmed by a detailed analysis of each cluster individually. By clicking on single clusters, a pop-up appears that does not just plot the single representative for the cluster but also the exact object values within the parallel coordinates plot.

After doing several of these interactions, the user can initiate a readjustment of the current cluster groupings (button 'adjust groupings'). As a result, refined and more sound concepts are identified. Technically, we realize the interactions and the regrouping by using constraint based clustering [WC00, WCRS01, Dav09]. The different types of interactions are implemented with particular must-link and cannot-link constraints between the subspace clusters.

The refined concepts, i.e., the novel grouping of clusters, cause new and refined significant dimensions for each concept. Accordingly, *CoDA* guides the user to the concept determination tab where novel thresholds within the bar charts can potentially be set, realizing a cyclic dependency between the determination and analysis of concepts to increase the quality of each step. By performing multiple iterations of this process, the user can gain a deeper understanding of the concept structure of large databases.

15.3 Exploring Multiple Clustering Solutions

15.3.1 Introduction

The detection of multiple clustering solutions is an active research area for traditional clustering as well as for subspace clustering. However, all of the presented methods focus just on the extraction of different concepts and their clusters. So far, there is no possibility to compare and analyze these alternative solutions. This, however, is the key for meaningful knowledge extraction. Are the concepts similar to each other or do they provide novel and interesting patterns? Do structures of one concept occur also in another one? Is this redundant information? What are the similarities or differences between individual clusters of multiple concepts? With MCExplorer (Multiple Concepts Explorer), we present a tool to close this gap through interactive exploration, browsing, and visualization of alternative clustering solutions. Starting with an overview of the entire structure, the user can progressively perform more fine-grained comparisons of patterns to

achieve an in-depth analysis. Focusing on the comparison of multiple concepts, and not on the individual concepts, MCExplorer comprises a three-level process that covers the whole cycle of analyzing multiple clusterings:

- Exploration of concepts, to compare multiple hidden groupings in the data.
- Exploration of clusters, to compare the clusters of different concepts.
- Exploration of elements, to compare the objects of different clusters.

With MCExplorer, the user can analyze alternative clustering solutions in an intuitive and interactive setting. Overall, MCExplorer supports the user in the knowledge extraction based on multiple valid groupings, completing the KDD process.

In the following subsections, we introduce MCExplorer that is integrated into the OpenSubspace [MAG+09a, MGAS09] and the CoDA (Section 15.2) framework, which add subspace clustering and multiple concept functionality to the well-known WEKA Data Mining Software. This framework provides MCExplorer with multiple groupings to be analyzed. Fig. 15.6 shows a screenshot of the framework with the MCExplorer integration.

The interactive exploration of MCExplorer is based on the Visual Exploration Paradigm [Kei02]: Starting with an overview of all concepts, the user can navigate through the visualization of these patterns and interesting concepts can be selected for a more detailed analysis. This detailed information can, again, be browsed and even more fine-grained information can be requested by the user. The comparison and analysis of multiple groupings, i.e., the coarsest level of the analysis, can be performed in the main window illustrated in Fig. 15.6, while the other two levels of MCExplorer are realized in child windows.

15.3.2 Exploring Concepts

At the start of the analysis, the user gets an overview of all multiple concepts, as shown in the main window of MCExplorer in Fig. 15.6. In this overview, each concept is represented as a node. Formally, each concept $Concept_i$ is a set of (subspace) clusters $\{C_1, \ldots, C_m\} = Concept_i$ with $C_j = (O_j, S_j)$ describing the grouped objects O_j and relevant dimensions S_j. Note that if a full-space clustering is analyzed, the set S_j is identical for each cluster and only the object grouping O_j is important. MCExplorer enables the user to quickly assess the concept structure by visualizing the core properties of single concepts and the relationships between different concepts. Core properties as the number of clustered objects

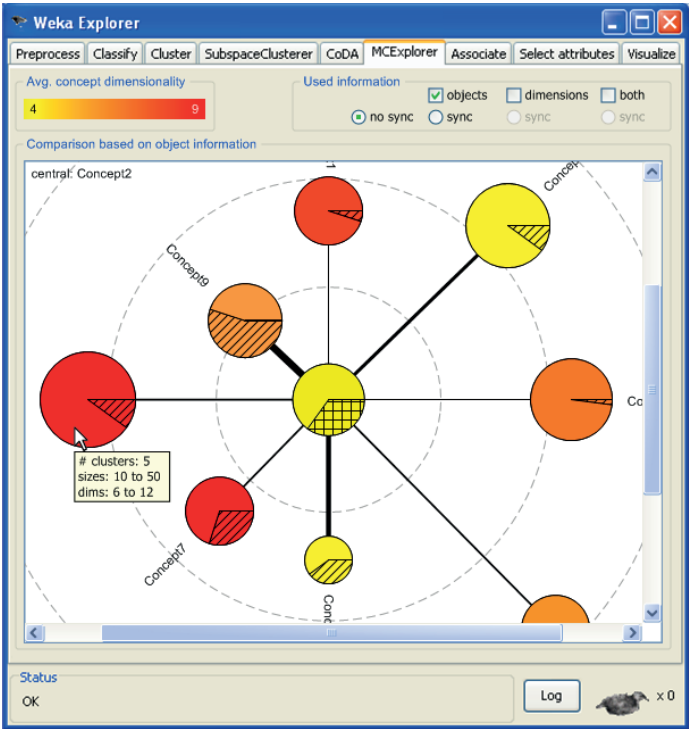


Figure 15.6: Main window of MCExplorer, integrated into WEKA.

and the average dimensionality are represented by the radius and color of a concept's corresponding node. More informations can be retrieved, when the cursor is placed over single nodes.

Several aspects are visualized that reflect the relationships between the different concepts. Initially, the user has to select a concept to be the current one under consideration (centrally displayed in Fig. 15.6). The remaining concepts are circularly arranged around this concept based on their similarity: Concepts very similar to the selected one are located near to the center while dissimilar concepts are at the border of the plot. Simultaneously, MCExplorer arranges the surrounding concepts along the circular lines such that similar concepts are adjacent. Technically, this is achieved by maximizing the pairwise similarity between adjacent concepts. When the user selects another concept as the central one, an

automatic rearrangement of the remaining concepts is performed. This visual approach enables the user to browse through the concept-structure and groups of (dis)-similar concepts can intuitively be captured. At each time, the user can select individual concepts to obtain a more detailed comparison between these; this detailed analysis of two concepts is described in Sec. 15.3.3. Formally, the similarity between the concepts is determined by the CE measure [PM06], which is applicable for subspace and full-space clustering solutions.

Each node in the concept structure is enriched with additional information with respect to the central concept. This enables a different kind of comparison in contrast to the similarity described above. More concretely, the pie-charts in the surrounding nodes indicate the degree to which a node's concept can be explained by using the central concept (the pie-chart of the central concept is explained later). If only few objects of the concept are also grouped in the same way in the central concept, the highlighted sector is small. If the sector is large, a redundancy of the concept is indicated, because the clusters of a concept can also be detected by the central concept. Formally, the fraction of a concept X that can be explained by another concept Y is determined via

$$explain(X|Y) = \frac{\sum_{C_j \in X} \max_{C_k \in Y} \{|O_j \cap O_k| \cdot |S_j \cap S_k|\}}{\sum_{C_j \in X} \{|O_j| \cdot |S_j|\}}$$

and, hence, the size of the sector for a surrounding concept Con given the central concept $Central$ is $explain(Con|Central)$. In contrast to the similarity between concepts, this information is non-symmetric. Therefore, MCExplorer visualizes the reverse property $explain(Central|Con)$ with the strength/weight of the edge connecting these two nodes, i.e., the edge weights indicate the degree to which the central concept can be explained by the other concepts. For the pie-chart of the central concept, we select the surrounding concept which explains most parts of the central one, indicating the highest degree of redundancy. Accordingly, the edge with the highest weight ($\max_{X \in Concepts}\{explain(Central|X)\}$) determines the size of the sector, where $Concepts$ is the set of all surrounding concepts.

The last feature on the level of exploring concepts allows the user to separately visualize the concepts just on the object information, just on the dimension information, or on both information together. Since relevant dimensions only occur in subspace clusterings, this visualization is constrained to inputs from subspace clustering algorithms. The obtainable knowledge is essential for the user, because two concepts can comprise identical object groupings but in com-

pletely different dimension sets. In MCExplorer, the user is able to choose which information to use for determining the similarity between the concepts and for the pie-chart calculation (cf. Fig. 15.6 top right corner). Overall, three different plots for the same central concept are possible and can be compared. However, since the arrangement of nodes is dynamically adjusted based on the similarities, different plots can show different layouts, which hinders an easy interpretation. To enable a visual comparison of these different plots, MCExplorer integrates a synchronization of the layouts. Thus, in MCExplorer, a single plot can be selected as the source, while the others are automatically synchronized, i.e., the central concept as well as the arrangement of adjacent nodes are taken over from the source plot.

15.3.3 Exploring Clusters

In the first level of MCExplorer, the user can obtain an overview of all concepts, and by selecting any two concepts within the plots a more detailed analysis can be performed. This analysis is done in the second level of MCExplorer, where the actual clusters of the selected concepts are compared, as illustrated in Fig. 15.7. If in the previous step two concepts are identified as similar, the user is now able to identify the causative clusters for this effect. For example, a cluster of one concept can be split up in smaller ones in another concept. To visually compare clusters, MCExplorer uses again an intuitive representation: clusters are represented by nodes where size and dimensionality are reflected by the node's radius and color. A horizontal ordering of the clusters is performed, such that similar clusters are placed in the same regions of the plot. This ordering is based on all clusters simultaneously and, hence, the user can easily compare the whole clustering structure of the two concepts. Redundancy between the clusters is indicated by a very close grouping of many clusters. The similarity between the clusters is formally defined by their overlapping elements and the ordering of clusters is obtained by minimizing the total weighted crossings in weighted bipartite graphs [cEKS09].

To increase the expressiveness of the visualization, the nodes are again enriched by pie-charts. As default, the visualized sector for each cluster indicates the overlap to its most similar cluster in the other concept (light shaded sectors in Fig. 15.7). For example, a cluster completely contained in another cluster of the other concept is enriched by a sector covering the whole node. This repre-

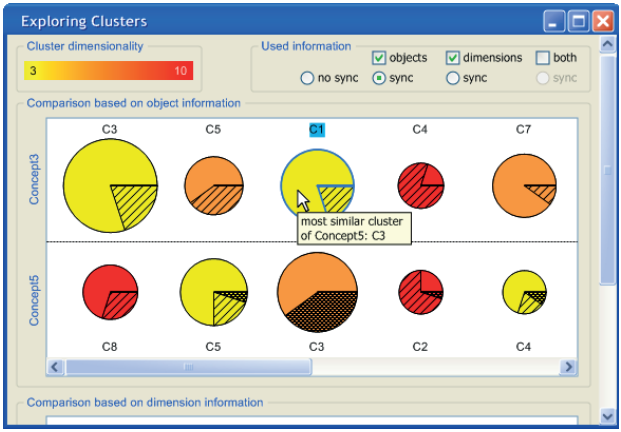


Figure 15.7: Detailed comparison of two concepts.

sentation is performed for each cluster of both concepts. With the help of this representation, several aspects, for example a cluster split up, can easily be detected. Furthermore, by selecting an individual cluster, its specific overlap with all clusters of the other concept can be analyzed (dark shaded sectors in Fig. 15.7). Analogously to the first level, the user can analyze objects and dimensions simultaneously or restrict the analysis to just one property.

Overall, the second level of MCExplorer enables the user to get an overview of the clustering structure of certain concepts and to understand the reasons for the similarity and dissimilarity of the detected clustering solutions.

15.3.4 Exploring Elements

The finest analysis in the comparison of multiple concepts can be done in the last level of MCExplorer: the individual elements of clusters can be compared by the user. By selecting clusters in the previous level, the user is guided to this level, as illustrated in Fig. 15.8. MCExplorer visualizes the data matrix and the embedded clusters. Each row corresponds to one object of the database and each column to one dimension. The elements contained in the selected clusters of the first concept are highlighted by rectangular regions within the matrix while the elements covered by clusters of the second concept are highlighted by circles. Overlapping and non-overlapping elements can, thus, visually be inspected

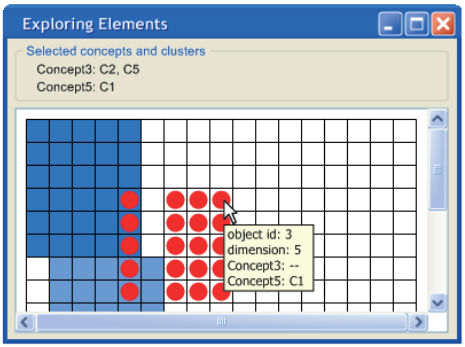


Figure 15.8: Comparison of concepts on the object level.

by the user. To facilitate this impression, MCExplorer permutes the objects and dimensions, such that the individual highlighted regions are mostly connected. Thus, outstanding elements which occur in several concepts can be identified. For a further in-depth analysis of such objects, the user is able to select arbitrary sets of elements in Fig. 15.8 and their characteristics, as the value-distributions within the dimensions, can be illustrated.

Overall, MCExplorer provides the user with the opportunity to compare multiple groupings in an interactive setting. Based on a three level process, the user is able to browse from an overview to an in-depth analysis: whole concepts, their corresponding clusters, and their individual elements can be compared and their properties can be analyzed. Besides the raw patterns determined by the existing methods, MCExplorer enables the user to infer actual knowledge based on the given clustering solutions.

15.4 Conclusion

In this chapter, we presented two tools for the visual analysis of subspace clustering results in the context of multi-view clustering. The first tool, CoDA, helps to determine and analyze concepts for the output of any subspace clustering method and to assign the detected subspace clusters to their corresponding concepts. Thereby, CoDA enables a transformation of any a subspace clustering result into a multi-view clustering result and, thus, helps in closing the gap between the two respective research areas.

The second tool, MCExplorer, focuses on the analysis of multi-view clustering results and especially those detected in subspace projections, e.g., subspace clustering results that have been post-processed with the CoDA tool, or the results of the presented approaches MVGen (Chapter 7), SMVC (Chapter 8), the graph-based framework of Part IV, or the approach presented in [NDJ10]. It supports the comparison and analysis of the generated alternative clustering solutions and, thereby, enables the user to decide on her preferences regarding different solutions but also to understand the different grouping patterns.

Part VI

Summary and Outlook

The sentence completes its signification only with its last term.
JAQUES LACAN

16

Conclusion and Future Work

The cluster analysis is a wide research field encompassing different techniques for different data types and different clustering goals. In this thesis, we focused on the detection of multiple, alternative clustering solutions for a single vector-represented dataset in different subspace projections. To this end, we joined the clustering paradigm of multi-view or alternative clustering, and the paradigm of subspace clustering. It was our major interest to point out the fundamental parallels between the two paradigms but also to highlight the different techniques to solve the two related problems. By combining both paradigms, we tried to exploit synergy effects of which each paradigm could benefit. In this chapter, we briefly review the main research contributions and highlight promising future research directions in this area.

16.1 Conclusion

In the first part of this thesis, we introduced the basic principles for the two paradigms of multi-view clustering and subspace clustering. We explained the effects of the curse of dimensionality, which hinder a meaningful cluster detection in the full-space. We also reviewed the related work for both paradigms, combined with a first rough categorization of the approaches. As both paradigms share the philosophy that instances of complex data might belong to multiple meaningful clusters and that different clustering views for the data exist, we presented approaches combining both principles in the following parts of this thesis.

The second part started with a thorough discussion of the relation between multi-view clustering and subspace clustering analyzing similarities and differences. The tackled challenge for this part was to improve the, so far, insufficient redundancy models of subspace clustering by using principles of multi-view clustering. Accordingly, this part took the perspective of subspace clustering and

integrated the idea of multiple clustering views into the clustering model. With the OSCLU model of Chapter 4, we proposed a global optimization of the overall clustering interestingness and used a redundancy model that evaluates the clusters' similarities regarding their sets of objects as well as their subspace projections. We have proven the optimization problem to be NP-hard and proposed an efficient algorithm to determine an approximate solution. In Chapter 5, we presented an extension of the OSCLU model to enable the integration of previously known subspace clusterings into the optimization process. The new clustering produced by ASCLU exhibits no redundancy to the a priori given clustering but complements its information to an overall optimal solution.

For the third part of this thesis, we took the perspective of multi-view approaches and realized different clustering perspectives on the data through different subspace projections and, thereby, incorporated the principles of subspace clustering. Compared to traditional subspace clustering, the aim was to explicitly determine multiple views, i.e., multiple partitionings of the data. For this part, we focused on the simultaneous generation of all clustering views to find an overall optimal set of clusterings. For our MVGen approach in Chapter 7, we introduced a novel generative model for data exhibiting multiple clusterings in different subspaces projections. The main focus for MVGen was to model classical subspace clustering scenarios, such as locally irrelevant dimensions for each cluster w.r.t. its view's subspace and shared relevant dimensions of different views. The model is designed for an arbitrary number of views and determines the relevant dimensions for each view and their subspace clusters by performing Bayesian model selection. For our generative model SMVC, presented in Chapter 8, we restricted the problem to a simpler subspace scenario, where views do not overlap regarding their relevant dimensions and where the relevant subspaces of clusters correspond to that of the respective view. This simplification allowed us to focus on the question of integrating partial information about the clustering structure as a priori given knowledge into the clustering process. The SMVC model allows such an integration of prior information via must-link and cannot-link instance level constraints and simultaneously learns the views, their clusterings, and the association of constraints to the views, such that the optimal clustering views are determined based on the provided instance level constraints.

In the fourth part of this thesis, we developed new techniques for the iterative detection of alternative clustering solutions in subspace projections. An iterative processing scheme allows to incorporate the knowledge of already present clus-

terings into the clustering process and, e.g., to steer into promising subspaces. In Chapter 9, we presented a graph-based framework, where previous clusterings are encoded as relational information of a graph structure. This allows the application of combined graph clustering methods for graphs annotated with feature vectors for the problem of finding a new clustering alternative. Since good alternative clusterings are located in subspace projections of the feature space, we presented two approaches for combined graph clustering in subspaces. Our SSCG algorithm of Chapter 10 is designed for graphs with multi-dimensional vertex labels and utilizes the principles of spectral clustering. To integrate the idea of subspace clustering, allowing each cluster to have an individual set of relevant dimensions, the affinity matrix for the eigendecomposition is part of the learning process of SSCG. To compare the clustering structures for different feature subsets, we defined an objective function that is unbiased regarding the number of relevant features. For graphs with multi-dimensional edge labels, we presented an extension of the popular modularity quality criterion for the subspace clustering paradigm in Chapter 11. We showed how existing optimization techniques can be applied for the new subspace modularity measure and presented a new clustering algorithm SuMo to deal more effectively with the problem of simultaneously determining a grouping and the subspaces based on our new subspace modularity measure.

The fifth part of this thesis concerns the evaluation and the visualization of subspace clustering results in the multi-view context. In Chapter 13, we collected four important requirements for the evaluation of subspace clustering results based on external evaluation measures, which equally apply for the evaluation of multi-view clusterings in subspace projections. We thoroughly examined the most frequently used evaluation measures w.r.t. these four criteria and presented the new E4SC measure to fulfill the necessary requirements for a fair evaluation of subspace clustering results. In Chapter 14, we proposed an interactive visualization workflow to support the choice of interesting subspaces for clustering if the processes of subspace search and clustering are decoupled. For the visualization and navigation of the different subspaces, we applied techniques for the subspace interestingness, the subspace alternativeness, and the clustering alternativeness and thereby combined the paradigms of subspace search, multi-view clustering, and subspace clustering. The tools that we presented in Chapter 15 support the user in defining and analyzing the views for a subspace clustering result. The CoDa tool provides an iterative, interactive process in which views

for a subspace clustering are suggested, analyzed, and potentially refined. It, thus, bridges the gap between subspace clustering and multi-view clustering by integrating users and their domain knowledge. The MCExplorer tool is designed to interactively explore and compare different clustering views to support a user in gaining insights into the semantic background of the clustering structure.

All presented techniques aimed at an alliance of the two related paradigms of subspace clustering and multi-view clustering. The developed techniques bridge the gap between the two paradigms and the experimental evaluations strongly support our claim, that meaningful alternative clusterings are to be found in subspace projections of the data. In several experiments, we demonstrated that existing techniques are outperformed by our novel methods.

16.2 Future Work

For the research areas of multi-view clustering and subspace clustering, we can identify further interesting and challenging research questions based on the results and models presented in this thesis. In recent publications, the research area of subspace clustering has contributed first techniques that transfer the principle of subspace clustering to various data domains to properly address the curse of dimensionality, e.g., time series data (e.g., [KGHS12]), stream data (e.g., [HKSS14]), or graph data (e.g., [GFBS14]). In the case of non-partitioning approaches, i.e., approaches that allow for overlapping clusters w.r.t. the object sets, these techniques need to take care of a proper redundancy handling. For their redundancy models the consideration of different concepts, as we proposed with our OSCLU model, will be a promising solution.

For this thesis, we restricted the consideration to vector data. However, since data is rarely collected pursuing only one defined analysis goal, the multi-view hypothesis is very likely for various other data domains as well. Subspace clustering methods might not be sufficient in such a multi-view scenario, as the evaluations of our approaches already indicated. In the small evaluation study for graph clustering methods of Chapter 12, we already encountered the impairment of the clustering results for an increasing number of hidden views. Given the size and complexity of the available graph data, e.g., for social networks, it is fair to assume that a single partitioning of the nodes is insufficient. Especially for graph clustering, the multi-view scenario is interesting as the different information sources, relations and vector annotations, might support different clustering

structures. Often the algorithms realize a trade-off between the density in the feature space and the relations [GFBS14]. Multi-view clustering can represent a different solution here, where not only subspaces but also information sources are emphasized for revealing a clustering view, which is especially interesting if simultaneously multi-dimensional edge labels and feature labels are considered.

For subspace clustering, designing efficient algorithms is one of the main challenges, which until now seems not to be solved sufficiently. While we applied our presented approaches on datasets with several thousand instances and tens of dimensions, databases of real-world applications often have millions of data entries and several hundreds of dimensions. Scalable algorithms for such large databases are indispensable, but also parallelization techniques need to be developed (e.g., [FWS14]).

With our semi-supervised SMVC approach (Chapter 8), we demonstrated the high potential of user provided prior knowledge regarding the clustering structure for improving the clustering quality in the multi-view scenario. It would be worth investigating how such prior information can be incorporated into the iterative processing scheme for detecting multiple alternatives in addition to the information provided by previously detected clusterings. Similarly, for subspace clustering in general, it is very promising to exploit prior information to enhance the quality of detected clustering solutions.

In this thesis, we presented first solutions to extend the multi-view clustering paradigm to the consideration of subspace projections. It would be worth to examine the possibilities to transfer further classical subspace clustering characteristics to the multi-view clustering paradigm, such as, e.g., view outliers (not every object must be part of a cluster in each view) or overlapping clusters within a single view (objects can fulfill different roles in a single view). A further challenging problem to address for the multi-view paradigm, is the question of how to estimate the number of views hidden in the data.

Part VII

Appendices

Derivation of Update Equation 7.U1

We first introduce two intermediate results.

BIC Approximation During our derivation we have to evaluate the term

$$\int_{\alpha}\int_{\beta} p(\alpha)p(\beta)\prod_{n\in I} Beta(X_{n,d};\alpha,\beta)\ \mathrm{d}\alpha\mathrm{d}\beta$$

where I is an index set denoting which observations are actually considered.

Using the Bayesian Information Criterion (BIC, or Schwarz criterion [Sch78, Bis06]), and the observation that in our case the Beta distribution is controlled by two free parameters, we can approximate the above term by

$$Beta_d(I) := \left[\prod_{n\in I} Beta(X_{n,d};\alpha_{MAP},\beta_{MAP})\right] \ / \ |I| \qquad \text{(A1)}$$

where α_{MAP} and β_{MAP} are the MAP estimates of the shape parameters given the set of observations.

Reformulating the Likelihood Let $N_{m,d} = \{n \in N \mid Dom_{n,d} = m\}$ be the set of all observations that choose the view m in dimension d as *dominant* and $N_{m,k,d} = \{n \in N_{m,d} \mid Sel_{n,m} = k\}$ be those observations which additionally *select* the cluster k. Obviously, for each dimension $d \in D$ we have $N = \bigcup_{m\in M}\bigcup_{k\in K} N_{m,k,d}$. Given the equation

$$\prod_{n\in N} \frac{V_{i(n),d}}{\sum_{m'\in M} V_{m',d}} \cdot p(X_{n,d} \mid Dom, Sel, S, \alpha, \beta)$$

where $i(n) = Dom_{n,d}$ denotes the dominant view of observation n. Instead of taking the product over each observation $n \in N$ individually, we can 'group' the observations according to their dominant view and selected cluster. That is, the above equation is equivalent to

$$\propto \prod_{m\in M}\left[\left(\frac{V_{m,d}}{\sum_{m'\in M} V_{m',d}}\right)^{|N_{m,d}|}\prod_{k\in K}\prod_{n\in N_{m,k,d}}\begin{cases} B_{k,d}^{n,m} & \text{if } S_{m,k,d}=1 \\ 1 & else \end{cases}\right] \qquad \text{(A2)}$$

and $B_{k,d}^{n,m}$ is an abbreviation for $Beta(X_{n,d};\alpha_{m,k,d},\beta_{m,k,d})$.

Derivation of Update Equation 7.U1 Using the above results, we now derive Equation 7.U1. We aim at maximizing

$$p(V_{m,d} \mid V \backslash \{V_{m,d}\}, Dom, Sel, \pi, X) \propto \sum_S \int_\alpha \int_\beta p(V, Dom, Sel, \pi, S, \alpha, \beta, X) \, \mathrm{d}\alpha \mathrm{d}\beta$$

Based on the dependencies given by our graphical model, this is equivalent to maximizing

$$\propto \sum_S \int_\alpha \int_\beta p(V) p(S \mid V) p(\alpha) p(\beta) p(Dom \mid S, Sel) p(X \mid Dom, Sel, S, \alpha, \beta) \, \mathrm{d}\alpha \mathrm{d}\beta$$

Making the individual variables explicit, we obtain

$$\propto \sum_S \int_\alpha \int_\beta \prod_{d \in D} \Bigg(\prod_{m \in M} \Big[p(V_{m,d}) \prod_{k \in K} \Big[p(S_{m,k,d} \mid V) p(\alpha_{m,k,d}) p(\beta_{m,k,d}) \Big] \Big] \cdot \prod_{n \in N} \Big[p(Dom_{n,d} \mid S, Sel) p(X_{n,d} \mid Dom, Sel, S, \alpha, \beta) \Big] \Bigg) \, \mathrm{d}\alpha \mathrm{d}\beta$$

Due to the summation over all possible realizations of S, the term $p(Dom_{n,d} \mid S, Sel)$ can well be approximated by the expected dominance of view $Dom_{n,d} =: i(n)$ in dimension d for observation n. The expected dominance is given by

$$\frac{V_{i(n),d}}{\sum_{m' \in M} V_{m',d}}$$

Thus, the above equation reformulates to

$$\propto \sum_S \int_\alpha \int_\beta \prod_{d \in D} \Bigg(\prod_{m \in M} \Big[p(V_{m,d}) \prod_{k \in K} \Big[p(S_{m,k,d} \mid V) p(\alpha_{m,k,d}) p(\beta_{m,k,d}) \Big] \Big] \cdot \underbrace{\prod_{n \in N} \Big[\frac{V_{i(n),d}}{\sum_{m' \in M} V_{m',d}} p(X_{n,d} \mid Dom, Sel, S, \alpha, \beta) \Big]}_{(*)} \Bigg) \, \mathrm{d}\alpha \mathrm{d}\beta$$

We now substitute part $(*)$ with the result obtained in Equation A2, and get after a reordering of the terms:

$$\propto \sum_S \int_\alpha \int_\beta \prod_{d\in D} \left(\prod_{m\in M} \left[p(V_{m,d}) \left(\frac{V_{m,d}}{\sum_{m'\in M} V_{m',d}} \right)^{|N_{m,d}|} \prod_{k\in K} \Big[p(S_{m,k,d} \mid V) \cdot \underbrace{p(\alpha_{m,k,d}) p(\beta_{m,k,d}) \prod_{n\in N_{m,k,d}} \begin{cases} B_{k,d}^{n,m} & \text{if } S_{m,k,d} = 1 \\ 1 & else \end{cases} \Big] \right] \right) \mathrm{d}\alpha \mathrm{d}\beta$$

$(**)$

The part $(**)$ can be substituted with the result obtained in Equation A1. Thus, the integration over α and β vanishes and the formula can further be simplified to:

$$\propto \sum_S \prod_{d\in D} \left(\prod_{m\in M} \left[p(V_{m,d}) \left(\frac{V_{m,d}}{\sum_{m'\in M} V_{m',d}} \right)^{|N_{m,d}|} \cdot \prod_{k\in K} \Big[p(S_{m,k,d} \mid V) \begin{cases} Beta_d(N_{m,k,d}) & \text{if } S_{m,k,d} = 1 \\ 1 & else \end{cases} \Big] \right] \right)$$

The summation over S disappears when making the two cases of $p(S_{m,k,d} \mid V)$ explicit, i.e. we introduce the two cases $p(S_{m,k,d} = 1 \mid V) = V_{m,d}$ and $p(S_{m,k,d} = 0 \mid V) = 1 - V_{m,d}$. Thus, the formula simplifies to:

$$\propto \prod_{d\in D} \left(\prod_{m\in M} \left[p(V_{m,d}) \left(\frac{V_{m,d}}{\sum_{m'\in M} V_{m',d}} \right)^{|N_{m,d}|} \cdot \prod_{k\in K} \left[V_{m,d} \cdot Beta_d(N_{m,k,d}) + (1 - V_{m,d}) \cdot 1 \right] \right] \right)$$

Let $c_d = \sum_{m'\in M, m'\neq m} V_{m',d}$. Since for the update of $V_{m,d}$ all remaining $V \backslash \{V_{m,d}\}$ are fixed, we have to maximize for each $V_{m,d}$:

$$p(V_{m,d}) \prod_{m'\in M} \left(\frac{V_{m',d}}{c_d + V_{m,d}} \right)^{|N_{m',d}|} \prod_{k\in K} \left[V_{m,d} \cdot Beta_d(N_{m,k,d}) + (1 - V_{m,d}) \cdot 1 \right]$$

Note that we still have to consider the terms $\left(\frac{V_{m',d}}{c_d + V_{m,d}} \right)$ for all $m' \in M$ since the variable $V_{m,d}$ appears in the denominator. Maximizing the above equation is

equivalent to maximizing

$$\propto p(V_{m,d}) \cdot V_{m,d}^{|N_{m,d}|} \cdot \prod_{m' \in M} (c_d + V_{m,d})^{-|N_{m',d}|} \prod_{k \in K} [V_{m,d} \cdot (Beta_d(N_{m,k,d}) - 1) + 1]$$

Since $\bigcup_{m' \in M} N_{m',d} = N$ and $V_{m,d}$'s prior follows a Beta distribution, we get 5

$$\propto V_{m,d}^{\alpha_{Rel}-1} \cdot (1 - V_{m,d})^{\beta_{Rel}-1} \cdot V_{m,d}^{|N_{m,d}|} \cdot (c_d + V_{m,d})^{-|N|} \cdot \prod_{k \in K} [V_{m,d} \cdot (Beta_d(N_{m,k,d}) - 1) + 1]$$

Taking the logarithm of the above equation and substituting $V_{m,d}$ with x yields

$$\propto (\alpha_{Rel} - 1 + |N_{m,d}|) \cdot \log x + (\beta_{Rel} - 1) \cdot \log(1 - x) + (-|N|) \cdot \log(x + c_d) + \sum_{k \in K} \log((Beta_d(N_{m,k,d}) - 1) \cdot x + 1)$$

Overall, the above equation corresponds to the update Equation 7.U1, which leads to a new realization for $V_{m,d}$.

$$V_{m,d} = \arg\max_{x \in (0,1)} c_a \cdot \log x + c_b \cdot \log(1 - x) + c_c \cdot \log(x + c_d) + \sum_{k \in K} \log(c_k \cdot x + 1)$$

where the c_* are constant values given by

$$c_a = \alpha_{Rel} - 1 + |N_{m,d}| \qquad c_b = \beta_{Rel} - 1 \qquad c_c = -|N|$$

$$c_d = \textstyle\sum_{m' \in M, m' \neq m} V_{m',d} \qquad c_k = Beta_d(N_{m,k,d}) - 1$$

Bibliography

[ABD+08] Elke Achtert, Christian Böhm, Jörn David, Peer Kröger, and Arthur Zimek. Global Correlation Clustering Based on the Hough Transform. *Statistical Analysis and Data Mining (SADM)*, 1(3):111–127, 2008.

[ABK+07a] Elke Achtert, Christian Böhm, Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. On Exploring Complex Relationships of Correlation Clusters. In *Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM)*, page 7, 2007.

[ABK+07b] Elke Achtert, Christian Böhm, Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Robust, Complete, and Efficient Correlation Clustering. In *Proceedings of the Seventh SIAM International Conference on Data Mining (SDM)*, 2007.

[AGAV09] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486, 2009.

[Agg04] Charu C. Aggarwal. A Human-Computer Interactive Method for Projected Clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(4):448–460, 2004.

[AGGR98] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD Conference)*, pages 94–105, 1998.

[AKMS07a] Ira Assent, Ralph Krieger, Emmanuel Müller, and Thomas Seidl. DUSC: Dimensionality Unbiased Subspace Clustering. In *Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM)*, pages 409–414, 2007.

[AKMS07b] Ira Assent, Ralph Krieger, Emmanuel Müller, and Thomas Seidl. VISA: visual subspace clustering analysis. *ACM SIGKDD Explorations Newsletter*, 9(2):5–12, 2007.

[AKMS08a] Ira Assent, Ralph Krieger, Emmanuel Müller, and Thomas Seidl. EDSC: Efficient density-based subspace clustering. In *Proceedings of the 17th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1093–1102, 2008.

[AKMS08b] Ira Assent, Ralph Krieger, Emmanuel Müller, and Thomas Seidl. INSCY: Indexing subspace clusters with in-process-removal of redundancy. In *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM)*, pages 719–724, 2008.

[AR10] Mohammad S. Aziz and Chandan K. Reddy. A Robust Seedless Algorithm for Correlation Clustering. In *Proceedings of the 14th Pacific-Asia Conferenceon Advances in Knowledge Discovery and Data Mining (PAKDD)*, pages 28–37, 2010.

[ATMF12] Leman Akoglu, Hanghang Tong, Brendan Meeder, and Christos Faloutsos. PICS: Parameter-free Identification of Cohesive Subgroups in Large Attributed Graphs. In *Proceedings of the 12th SIAM International Cpnference on Data Mining (SDM)*, pages 439–450, 2012.

[AWY+99] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia M. Procopiuc, and Jong Soo Park. Fast Algorithms for Projected Clustering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 61–72, 1999.

[AY00a] Charu C. Aggarwal and Philip S. Yu. Finding Generalized Projected Clusters In High Dimensional Spaces. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD Conference)*, pages 70–81, 2000.

[AY00b] Charu C. Aggarwal and Philip S. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD Conference)*, pages 70–81, 2000.

[BB06] Eric Bae and James Bailey. COALA : A Novel Approach for the Extraction of an Alternate Clustering of High Quality and High Dissimilarity. In *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM)*, pages 53–62, 2006.

[BBD10] Eric Bae, James Bailey, and Guozhu Dong. A clustering comparison measure using density profiles and its application to the discovery of alternate clusterings. *Data Mining and Knowledge Discovery (DMKD)*, 21(3):427–471, 2010.

[BBM04a] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Active Semi-Supervision for Pairwise Constrained Clustering. In *Proceedings of the Fourth SIAM International Conference on Data Mining (SDM)*, pages 333–344, 2004.

[BBM04b] Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. A Probabilistic Framework for Semi-Supervised Clustering. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 59–68, 2004.

[BBM04c] 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, 2004.

[BCG11] Michele Berlingerio, Michele Coscia, and Fosca Giannotti. Finding and Characterizing Communities in Multidimensional Networks. In *IEEE International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 490–494, 2011.

[BDG+06] Ulrik Brandes, Daniel Delling, Marco Gaertler, Görke Robert, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity-np-completeness and beyond. Technical report, 2006.

[BDW08] Sugato Basu, Ian Davidson, and Kiri Wagstaff. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC, 2008.

[Bes86] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48(3):259–302, 1986.

[BGHS12] Brigitte Boden, Stephan Günnemann, Holger Hoffmann, and Thomas Seidl. Mining Coherent Subgraphs in Multi-Layer Graphs

with Edge Labels. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1258–1266, 2012.

[BGHS13] Brigitte Boden, Stephan Günnemann, Holger Hoffmann, and Thomas Seidl. RMiCS: A Robust Approach for Mining Coherent Subgraphs in Edge-Labeled Multi-Layer Graphs. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management (SSDBM)*, pages 23:1–23:12, 2013.

[BGLL08] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[BGRS99] Kevin S. Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When Is "Nearest Neighbor" Meaningful? In *Proceedings of the Seventh International Conference on Database Theory (ICDT)*, pages 217–235, 1999.

[Bis06] Christopher M. Bishop. *Pattern recognition and machine learning (Information Science and Statistics)*, volume 4. Springer, New York, 2006.

[BJ06] Francis R. Bach and Michael I. Jordan. Learning Spectral Clustering, With Application To Speech Separation. *Journal of Machine Learning Research (JMLR)*, 7:1963–2001, 2006.

[BKG+05] Arindam Banerjee, Chase Krumpelman, Joydeep Ghosh, Sugato Basu, and Raymond J. Mooney. Model-based overlapping clustering. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 532–537, 2005.

[BKKZ04] Christian Böhm, Karin Kailing, Peer Kröger, and Arthur Zimek. Computing Clusters of Correlation Connected Objects. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD Conference)*, pages 455–466, 2004.

[BKM+13] Klemens Böhm, Fabian Keller, Emmanuel Müller, Hoang Vu Nguyen, and Jilles Vreeken. CMI: An Information-Theoretic Contrast Measure for Enhancing Subspace Cluster and Outlier Detection. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM)*, pages 198–206, 2013.

[Bod14] Brigitte Boden. *Combined clustering of graph and attribute data*. PhD thesis, RWTH Aachen University, 2014.

[BPK+04] Christian Baumgartner, Claudia Plant, Karin Kailing, Hans-Peter Kriegel, and Peer Kröger. Subspace Selection for Clustering High-Dimensional Data. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM)*, pages 11–18. IEEE CS Press, 2004.

[Bre73] Richard P. Brent. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, New Jersey, 1973.

[BT78] Richard J. Beckman and Garth L. Tietjen. Maximum likelihood estimation for the beta distribution. *Journal of Statistical Computation and Simulation*, 7(3-4):253–258, 1978.

[BvLBS11] Sebastian Bremm, Tatiana von Landesberger, Jürgen Bernard, and Tobias Schreck. Assisted Descriptor Selection Based on Visual Comparative Data Analysis. *Computer Graphics Forum*, 30(3):891–900, 2011.

[BvLH+11] Sebastian Bremm, Tatiana von Landesberger, Martin Heß, Philipp Weil, and Kay Hamacher. Interactive Visual Comparison of Multiple Trees. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 31–40. IEEE CS Press, 2011.

[BZ07] Björn Bringmann and Albrecht Zimmermann. The Chosen Few: On Identifying Valuable Patterns. In *Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM)*, pages 63–72, 2007.

[CC94] Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling*. Chapman & Hall, 1994.

[cEKS09] Olca A. Çakiroğlu, Cesim Erten, Ömer Karataş, and Melih Sözdinler. Crossing minimization in weighted bipartite graphs. *Journal of Discrete Algorithms*, 7(4):439–452, 2009.

[CENS06] Rich Caruana, Mohamed F. Elhawary, Nam Nguyen, and Casey Smith. Meta Clustering. In *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM)*, pages 107–118, 2006.

[CFD07] Ying Cui, Xiaoli Z. Fern, and Jennifer G. Dy. Non-redundant Multiview Clustering via Orthogonalization. In *Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM)*, pages 133–142, 2007.

[CFZ99] Chun-Hung Cheng, Ada Waichee Fu, and Yi Zhang. Entropy-based subspace clustering for mining numerical data. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 84–93, 1999.

[CGSQ11] Nan Cao, David Gotz, Jimeng Sun, and Huamin Qu. DICON: Interactive Visual Analysis of Multidimensional Clusters. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 17(12):2581–2590, 2011.

[CK01] Anne Condon and Richard M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures Algorithms*, 18(2):116–140, 2001.

[CNM04] Aaron Clauset, Mark E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111, 2004.

[CT02] Gal Chechik and Naftali Tishby. Extracting Relevant Structures with Side Information. In *Advances in Neural Information Processing Systems 15 (NIPS)*, pages 857–864, 2002.

[CZ09] Yong Cheng and Ruilian Zhao. Multiview spectral clustering via ensemble. In *IEEE International Conference on Granular Computing (CRC)*, pages 101–106, 2009.

[Dav09] Ian Davidson. Clustering with Constraints. In Ling Liu and Tamer M. Özsu, editors, *Encyclopedia of Database Systems*, pages 393–396. Springer US, 2009.

[Dav12] Ian Davidson. Two Approaches to Understanding when Constraints Help Clustering. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1312–1320, 2012.

[DB10a] Xuan H. Dang and James Bailey. A Hierarchical Information Theoretic Technique for the Discovery of Non Linear Alternative Clusterings. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 573–582, 2010.

[DB10b] Xuan Hong Dang and James Bailey. Generation of Alternative Clusterings Using the CAMI Approach. In *Proceedings of the Tenth SIAM International Conference on DataMining (SDM)*, pages 118–129, 2010.

[DB13a] Xuan H. Dang and James Bailey. A Framework to Uncover Multiple Alternative Clusterings. *Machine Learning*, pages 1–24, 2013.

[DB13b] Xuan H. Dang and James Bailey. Generating Multiple Alternative Clusterings Via Globally Optimal Subspaces. *Data Mining and Knowledge Discovery (DMKD)*, 28(3):569–592, 2013.

[DFVN12] Xiaowen Dong, Pascal Frossard, Pierre Vandergheynst, and Nikolai Nefedov. Clustering With Multi-Layer Graphs: A Spectral Perspective. *IEEE Transactions on Signal Processing*, 60(11):5820 –5831, 2012.

[DQ08] Ian Davidson and Zijie Qi. Finding Alternative Clusterings Using Constraints. In *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM)*, pages 773–778, 2008.

[EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining (KDD), pages 226–231, 1996.

[FA10] Andrew Frank and Arthur Asunction. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2010.

[FB08] Qiang Fu and Arindam Banerjee. Multiplicative Mixture Models for Overlapping Clustering. In *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM)*, pages 791–796, 2008.

[FBT+10] Bilkis J. Ferdosi, Hugo Buddelmeijer, Scott C. Trager, Michael H. F. Wilkinson, and Jos B. T. M. Roerdink. Finding and visualizing relevant subspaces for clustering high-dimensional astronomical data using connected morphological operators. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 35–42. IEEE CS Press, 2010.

[For10] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[FWS14] Sergej Fries, Stephan Wels, and Thomas Seidl. Projected clustering for huge data sets in mapreduce. In *Proceedings of the 17th International Conference on Extending Database Technology (EDBT)*, pages 49–60, 2014.

[GBS12a] Stephan Günnemann, Brigitte Boden, and Thomas Seidl. Finding density-based subspace clusters in graphs with feature vectors. *Data Mining and Knowledge Discovery (DMKD)*, 25(2):243–269, 2012.

[GBS12b] Stephan Günnemann, Brigitte Boden, and Thomas Seidl. Finding density-based subspace clusters in graphs with feature vectors. *Data Mining and Knowledge Discovery (DMKD)*, 25(2):243–269, September 2012.

[GEG+08] Rong Ge, Martin Ester, Byron J. Gao, Zengjian Hu, Binay Bhattacharya, and Boaz Ben-Moshe. Joint cluster analysis of attribute data and relationship data: The connected k-center problem, algorithms and applications. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):1–35, 2008.

[Gen02] Marc G. Genton. Classes of kernels for machine learning: a statistics perspective. *Journal of Machine Learning Research (JMLR)*, 2:299–312, 2002.

[GFBS10] Stephan Günnemann, Ines Färber, Brigitte Boden, and Thomas Seidl. Subspace Clustering Meets Dense Subgraph Mining: A Synthesis of Two Paradigms. In *Proceedings of the Tenth IEEE International Conference on Data Mining (ICDM)*, pages 845–850, 2010.

[GFBS14] Stephan Günnemann, Ines Färber, Brigitte Boden, and Thomas Seidl. GAMer: A Synthesis of Subspace Clustering and Dense Subgraph Mining. *Knowledge and Information Systems (KAIS)*, 40(2):243–278, 2014.

[GFKS10] Stephan Günnemann, Ines Färber, Hardy Kremer, and Thomas Seidl. CoDA : Interactive Cluster Based Concept Discovery. *Proceedings of the VLDB Endowment*, 3(2):1633–1636, 2010.

[GFM+11] Stephan Günnemann, Ines Färber, Emmanuel Müller, Ira Assent, and Thomas Seidl. External Evaluation Measures for Subspace Clustering. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1363–1372, 2011.

[GFMS10] Stephan Günnemann, Ines Färber, Emmanuel Müller, and Thomas Seidl. ASCLU: Alternative Subspace Clustering. In *First Workshop on Discovering, Summarizing, and Using Multiple Clusterings (MultiClust) in conjunction with the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Minng (KDD)*, 2010.

[GFRS13] Stephan Günnemann, Ines Färber, Sebastian Raubach, and Thomas Seidl. Spectral Subspace Clustering for Graphs with Feature Vectors. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM)*, pages 231–240. IEEE Computer Society, 2013.

[GFRS14] Stephan Günnemann, Ines Färber, Matthias Rüdiger, and Thomas Seidl. SMVC: Semi-Supervised Multi-View Clustering in Subspace Projections. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[GFS12] Stephan Günnemann, Ines Färber, and Thomas Seidl. Multi-View Clustering Using Mixture Models in Subspace Projections. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 132–140, 2012.

[GFVS12] Stephan Günnemann, Ines Färber, Kittipat Virochsiri, and Thomas Seidl. Subspace Correlation Clustering: Finding Locally Correlated Dimensions in Subspace Projections of the Data. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 352–360, 2012.

[GH03] David Gondek and Thomas Hofmann. Conditional Information Bottleneck Clustering. In *Workshop on Clustering Large Data Sets, in Conjunction with the Third IEEE International Conference on Data Mining (ICDM)*, pages 36–42, 2003.

[GH04] David Gondek and Thomas Hoffmann. Non-Redundant Data Clustering. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM)*, pages 75–82, 2004.

[GH05] David Gondek and Thomas Hofmann. Non-Redundant Clustering with Conditional Ensembles. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 70–77, 2005.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, San Francisco, 1979.

[GKFS10] Stephan Günnemann, Hardy Kremer, Ines Färber, and Thomas Seidl. MCExplorer: Interactive Exploration of Multiple (Subspace) Clustering Solutions. In *Proceedings of the IEEE International Conference on Data Mining Workshops(ICDMW)*, pages 1387–1390, Washington,USA, 2010. IEEE Computer Society.

[GMFS09] Stephan Günnemann, Emmanuel Müller, Ines Färber, and Thomas Seidl. Detection of Orthogonal Concepts in Subspaces of High Dimensional Data. In *Proceedings of the 18th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1317–1326, 2009.

[GVG05] David Gondek, Shivakumar Vaithyanathan, and Ashutosh Garg. Clustering with Model-Level Constraints. In *Proceedings of the Fifth SIAM International Conference on Data Mining (SDM)*, pages 126–137, 2005.

[HH07] Rave Harpaz and Robert M. Haralick. Mining Subspace Correlations. In *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 335–342, 2007.

[HKK+10] Michael E. Houle, Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Can Shared-Neighbor Distances Defeat the Curse of Dimensionality? In *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management (SSDBM)*, pages 482–500, 2010.

[HKP11] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 3rd edition, 2011.

[HKSS14] Marwan Hassani, Philipp Kranen, Rajveer Saini, and Thomas Seidl. Subspace anytime stream clustering. In *Proceedings of the 25th International Conference on Scientific and Statistical Database (SSDBM)*, page 37, 2014.

[HTW+10] M. Shahriar Hossain, Satish Tadepalli, Layne T. Watson, Ian Davidson, Richard F. Helm, and Naren Ramakrishnan. Unifying dependent clustering and disparate clustering for non-homogeneous data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 593–602, 2010.

[HZZL02] Daniel Hanisch, Alexander Zien, Ralf Zimmer, and Thomas Lengauer. Co-clustering of biological networks and gene expression data. *Proceedings of the Tenth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 18(1):145–154, 2002.

[ID90] Alfred Inselberg and Bernard Dimsdale. Parallel Coordinates: A Tool for Visualizing Multi-dimensional Geometry. In *Proceedings of the First IEEE Conference on Visualization*, pages 361–378, 1990.

[IMI+10] Stephen Ingram, Tamara Munzner, Veronika Irvine, Melanie Tory, Steven Bergner, and Thorsten Möller. DimStiller: Workflows for dimensional analysis and reduction. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 3–10. IEEE CS Press, 2010.

[JD88] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., NJ, USA, 1988.

[JJ09] Sara Johansson and Jimmy Johansson. Interactive Dimensionality Reduction Through User-defined Combinations of Quality Metrics. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 15(6):993–1000, 2009.

[JMD08] Prateek Jain, Raghu Meka, and Inderjit S. Dhillon. Simultaneous Unsupervised Learning of Disparate Clusterings. *Statistical Analysis and Data Mining*, 1(3):195–210, 2008.

[Jol02] Ian T. Jolliffe. *Principal Components Analysis*. Springer, 2nd edition, 2002.

[Kan92] Viggo Kann. *On the approximability of NP-complete optimization problems*. PhD thesis, Royal Institute of Technology Stockholm, 1992.

[KdB13] Kleanthis-Nikolaos Kontonasios and Tijl de Bie. Subjectively interesting alternative clusterings. *Machine Learning*, pages 1–26, 2013.

[Kei02] Daniel A. Keim. Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 8(1):1–8, 2002.

[KGHS12] Hardy Kremer, Stephan Günnemann, Arne Held, and Thomas Seidl. Effective and robust mining of temporal subspace clusters. In *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM)*, pages 369–378, 2012.

[KKK04] Karin Kailing, Hans-Peter Kriegel, and Peer Kröger. Density-connected subspace clustering for high-dimensional data. In *Proceedings of the Fourth SIAM International Conference on Data Mining (SDM)*, pages 246–256, 2004.

[KKKW03] Karin Kailing, Hans-Peter Kriegel, Peer Kröger, and Stefanie Wanka. Ranking Interesting Subspaces for Clustering High Dimensional Data. In *Proceedings of the Seventh European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 241–252, 2003.

[KKZ09] 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), 2009.

[KL70] Brian W. Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, 49(2):291–307, 1970.

[KMB12] Fabian Keller, Emmanuel Müller, and Klemens Böhm. HiCS: High Contrast Subspaces for Density-Based Outlier Ranking. In *Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE)*, pages 1037–1048, 2012.

[KMWB13] Fabian Keller, Emmanuel Müller, Andreas Wixler, and Klemens Böhm. Flexible and adaptive subspace search for outlier analysis. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1381–1390, 2013.

[KRDI10] Abhishek Kumar, Piyush Rai, and Hal Daumé III. Co-regularized spectral clustering with multiple kernels. In *Advances in Neural Information Processing Systems 23 (NIPS) Workshop: New Directions in Multiple Kernel Learning*, 2010.

[LL04] Zhengdong Lu and Todd K. Leen. Semi-supervised learning with penalized probabilistic clustering. In *Advances in Neural Information Processing Systems 17 (NIPS)*, pages 849–856, 2004.

[LLX+10] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of Internal Clustering Validation Measures. In *Proceedings of the Tenth IEEE International Conference on Data Mining (ICDM)*, pages 911–916, 2010.

[LM07] Huan Liu and Hiroshi Motoda. *Computational Methods of Feature Selection*. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC, 2007.

[LSP+10] Alexander Lex, Marc Streit, Christian Partl, Karl Kashofer, and Dieter Schmalstieg. Comparative Analysis of Multidimensional, Quantitative Data. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 16(6):1027–1035, 2010.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[MAG+09a] Emmanuel Müller, Ira Assent, Stephan Günnemann, Timm Jansen, and Thomas Seidl. OpenSubspace: An Open Source Framework for Evaluation and Exploration of Subspace Clustering Algorithms in WEKA. In *Proceedings of the First Open Source in Data Mining Workshop (OSDM) in conjunction with the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 2–13, 2009.

[MAG+09b] Emmanuel Müller, Ira Assent, Stephan Günnemann, Ralph Krieger, and Thomas Seidl. Relevant subspace clustering: Mining the most interesting non-redundant concepts in high dimensional data. In *Proceedings of the Ninth IEEE International Conference on Data Mining (ICDM)*, pages 377–386, 2009.

[MCP08] Rafael Martí, Vicente Campos, and Estefanía Piñana. A branch and bound algorithm for the matrix bandwidth minimization. *European Journal of Operational Research*, 186(2):513–528, 2008.

[MCRE09] Flavia Moser, Recep Colak, Arash Rafiey, and Martin Ester. Mining cohesive patterns from graphs with feature vectors. In *Proceedings of the Ninth SIAM International Conference on DataMining (SDM)*, pages 593–604, 2009.

[Mey00] Carl D. Meyer. *Matrix analysis and applied linear algebra book and solutions manual*, volume 2. SIAM: Society for Industrial and Applied Mathematics, Philadelphia, 2000.

[MGAS09] Emmanuel Müller, Stephan Günnemann, Ira Assent, and Thomas Seidl. Evaluating Clustering in Subspace Projections of High Dimensional Data. *Proceedings of the VLDB Endowment*, 2(1):1270–1281, 2009.

[MGFS10] Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Proceedings of the Tenth IEEE International Conference on Data Mining (ICDM)*, page 1220, 2010.

[MK08] Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*. Wiley series in probability and statistics. Wiley, 2nd edition, 2008.

[MS00] Erkki Mäkinen and Harri Siirtola. Reordering the Reorderable Matrix as an Algorithmic Problem. In *Proceedings of the First International Conference on Theory and Application of Diagrams (Diagrams)*, pages 453–467, 2000.

[MS01] Marina Meila and Jianbo Shi. A Random Walks View of Spectral Segmentation. In *Proceedings of the Fourth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2001.

[MS08] Gabriela Moise and Jörg Sander. Finding non-redundant, statistically significant regions in high dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 533–541, 2008.

[MSE06] Gabriela Moise, Jörg Sander, and Martin Ester. P3C: A Robust Projected Clustering Algorithm. In *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM)*, pages 414–425, 2006.

[MZK+09] Gabriela Moise, Arthur Zimek, Peer Kröger, Hans-Peter Kriegel, and Jörg Sander. Subspace and projected clustering: experimental evaluation and analysis. *Knowledge and Information Systems (KAIS)*, 21(3):299–326, 2009.

[NDJ10] Donglin Niu, Jennifer G. Dy, and Michael I. Jordan. Multiple Non-Redundant Spectral Clustering Views. In *Proceedings of the 27th*

International Conference on Machine Learning (ICML), pages 831–838, 2010.

[New04a] Mark E. J. Newman. Analysis of weighted networks. *Physical review E*, 70(5):056131, 2004.

[New04b] Mark E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[New06] Mark E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[NG04] Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, 2004.

[NGC01] Harsha S. Nagesh, Sanjay Goil, and Alok N. Choudhary. Adaptive grids for clustering massive data sets. In *Proceedings of the First SIAM International Conference on Data Mining (SDM)*, 2001.

[NMB13] Hoang Vu Nguyen, Emmanuel Müller, and Klemens Böhm. 4S: Scalable subspace search scheme overcoming traditional Apriori processing. In *Proceedings of the 2013 IEEE International Conference on Big Data*, pages 359–367, 2013.

[PHL04] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: a review. *ACM SIGKDD Explorations Newsletter*, 6(1):90–105, 2004.

[PJAM02] Cecilia M. Procopiuc, Michael Jones, Pankaj K. Agarwal, and T. M. Murali. A Monte Carlo algorithm for fast projective clustering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD Conference)*, pages 418–427, 2002.

[PM06] Anne Patrikainen and Marina Meila. Comparing subspace clusterings. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 18(7):902–916, 2006.

[PZCW10] Leonard K. M. Poon, Nevin Lianwen Zhang, Tao Chen, and Yi Wang. Variable Selection in Model-Based Clustering: To Do or To Facilitate. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 887–894, 2010.

[QAH12] Guo-Jun Qi, Charu C. Aggarwal, and Thomas S. Huang. Community Detection with Edge Content in Social Media Networks. In *Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE)*, pages 534–545, 2012.

[QD09] ZiJie Qi and Ian Davidson. A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 717–726, 2009.

[Ran71] William M Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[RT60] David J. Rogers and Taffee T. Tanimoto. A Computer Program for Classifying Plants. *Science*, 132(3434):1115–1118, 1960.

[SBHHW03] Noam Shental, Aharon Bar-Hillel, Tomer Hertz, and Daphna Weinshall. Computing Gaussian Mixture Models with EM Using Equivalence Constraints. In *Advances in Neural Information Processing Systems 16 (NIPS)*, pages 465–472, 2003.

[Sch78] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[Sch07] Satu E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages (VL)*, pages 336–343. IEEE CS Press, 1996.

[Shn02] Ben Shneiderman. Interactively Exploring Hierarchical Clustering Results. *IEEE Computer*, 35(7):80–86, 2002.

[Sil86] Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

[SJR10] Manas Somaiya, Christopher M. Jermaine, and Sanjay Ranka. Mixture models for learning low-dimensional roles in high-dimensional data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 909–918, 2010.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):888–905, 2000.

[SS04] Jinwook Seo and Ben Shneiderman. A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis)*, pages 65–72, 2004.

[ST96] Daniel A. Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 96–105, 1996.

[STM07] Motoki Shiga, Ichigatu Takigawa, and Hiroshi Mamitsuka. A spectral clustering approach to optimally combining numericalvectors with a modular network. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 647–656, 2007.

[SvLB10] Tobias Schreck, Tatiana von Landesberger, and Sebastian Bremm. Techniques for Precision-Based Visual Analysis of Projected Data. *Information Visualization*, 9(3):181–193, 2010.

[SZ04] Karlton Sequeira and Mohammed Zaki. SCHISM: A New Approach for Interesting Subspace Mining. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM)*, pages 186–193, 2004.

[TLD09] Wei Tang, Zhengdong Lu, and Inderjit S. Dhillon. Clustering with multiple graphs. In *Proceedings of the Ninth IEEE International Conference on Data Mining (ICDM)*, pages 1016–1021, 2009.

[TMF+12] Andrada Tatu, Fabian Maaß, Ines Färber, Enrico Bertini, Tobias Schreck, Thomas Seidl, and Daniel A. Keim. Subspace Search and Visualization to Make Sense of Alternative Clusterings in High-Dimensional Data. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 63–72. IEEE Computer Society, 2012.

[TPB99] Naftali Tishby, Fernendo C. Pereira, and William Bialek. The information bottleneck method. In *Proeedings of the 37th Annual Allerton Confernece on Communication, Control, and Computing*, pages 368–377. The Computing Research Repository (CoRR), 1999.

[TWL12] Lei Tang, Xufei Wang, and Huan Liu. Community detection via heterogeneous interaction analysis. *Data Mining and Knowledge Discovery (DMKD)*, 25(1):1–33, 2012.

[vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[VE10] Nguyen X. Vinh and Julien Epps. minCEntropy: A Novel Information Theoretic Approach for the Generation of Alternative Clusterings. In *Proceedings of the Tenth IEEE International Conference on Data Mining (ICDM)*, pages 521–530, December 2010.

[VEB10] Nguyen X. Vinh, Julien Epps, and James Bailey. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *Journal of Machine Learning Research (JMLR)*, 11:2837–2854, 2010.

[VK09] Soujanya Vadapalli and Kamalakar Karlapalem. Heidi matrix: nearest neighbor driven high dimensional data visualization. In *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery (VAKD): Integrating Automated Analysis with Interactive Exploration*, pages 83–92, 2009.

[vL07] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[WAG05] Leland Wilkinson, Anushka Anand, and Robert Grossman. Graph-Theoretic Scagnostics. In *IEEE Symposium on Information Visualization (InfoVis)*, pages 157–164. IEEE CS Press, 2005.

[WC00] Kiri Wagstaff and Claire Cardie. Clustering with Instance-level Constraints. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 1103–1110, 2000.

[WCRS01] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained K-means Clustering with Background Knowledge. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 577–584, 2001.

[WGK10] Matthew Ward, Georges G. Grinstein, and Daniel Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications*. A. K. Peters, Ltd. / CRC Press, Natick, MA, USA, 2010.

[WJ63] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

[WXC09] 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.

[YHJ09] Donghui Yan, Ling Huang, and Michael I. Jordan. Fast approximate spectral clustering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 907–916, 2009.

[YM03] Man L. Yiu and Nikos Mamoulis. Frequent-Pattern based Iterative Projected Clustering. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM)*, pages 689–692, 2003.

[YPH+04] Jing Yang, Anilkumar Patro, Shiping Huang, Nishant K. Mehta, Matthew O. Ward, and Elke A. Rundensteiner. Value and Relation

Display for Interactive Exploration of High Dimensional Datasets. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis)*, pages 73–80. IEEE CS Press, 2004.

[YWG11] Xiaoru Yuan, Zuchao Wang, and Cong Guo. MDS-Tree and MDS-Matrix for High Dimensional Data Visualization. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis)*, 2011.

[YWRH03] Jing Yang, Matthew O. Ward, Elke A. Rundensteiner, and Shiping Huang. Visual hierarchical dimension reduction for exploration of high dimensional datasets. In *Proceedings of the IEEE Symposium on Visualization (VISSYM)*, pages 19–28. Eurographics Association, 2003.

[ZCY09] Yang Zhou, Hong Cheng, and Jeffrey X. Yu. Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment*, 2(1):718–729, January 2009.

Statement Of Originality

Many of the ideas and techniques presented in this thesis evolved from numerous fruitful discussions within the group of Professor Seidl. The high level of collaboration with my colleagues, students, and project partners provided a very productive and inspiring environment, making it hard to pinpoint individual contributions in detail. The following provides some more information regarding collaborations and support for the individual chapters. Parts I, VI, and VII, as well as Chapters 3, 6, 9, and 12 constitute new content, added in order to relate and to introduce the main parts of this thesis. The OSCLU method, presented in Chapter 4, was developed, implemented, and evaluated by me during my diploma thesis, which has been supervised by Emmanuel Müller and Stephan Günnemann. The extension of OSCLU to the ASCLU approach in Chapter 5 has also been developed, implemented, and evaluated by me. For OSCLU, as well as ASCLU Emmanuel Müller and Stephan Günnemann helped much with the formalization and presentation of the final ideas which led to the publications [GMFS09, GFMS10]. The MVGen model, presented in Chapter 7, was developed in collaboration with Stephan Günnemann and a preliminary implementation was done by Patrick Gerwert during his master thesis. The contents of Chapter 7 were published in [GFS12]. The SMVC model in Chapter 8 has been developed in collaboration with Stephan Günnemann and Matthias Rüdiger, who implemented the SMVC model during his master thesis. The experimental evaluation of the SMVC model was conducted by me and the contents of Chapter 8 were published in [GFRS14]. The SSCG clustering method presented in Chapter 10 was implemented and evaluated by Sebastian Raubach in his master thesis, whom I advised together with Stephan Günnemann. The SSCG algorithm was published in [GFRS13]. The SUMO approach of Chapter 11 was developed in the bachelor thesis of Grzegorz Stepien, whom I advised together with Brigitte Boden. The SUMO approach was published in [Bod14]. The E4SC measure in Chapter 13 was developed in collaboration with Emmanuel Müller and Stephan Günnemann and was published in [GFM+11]. Most of the evaluation was conducted by Patrick Gerwert. The contents of Chapter 14 were published in [TMF+12] and are the result of a strong collaboration with Andrada Tatu, Fabian Maaß, and Enrico Bertini within the funded research project SteerSCiVA DFG-664/11. The tools Coda and MCExplorer presented in Chapter 15 were published in [GFKS10] and [GKFS10]. The implementation of both was mostly realized by Matthias Hannen.

List of Publications

1. Christian Böhm, Ines Färber, Sergej Fries, Ulrike Korte, Johannes Merkle, Annahita Oswald, Thomas Seidl, Bianca Wackersreuther, and Peter Wackersreuther. Filtertechniken für geschützte biometrische Datenbanken. In *Proceedings of the 14th GI Conference on Database Systems for Business, Technology, and the Web (BTW)*, pages 379–389, 2011.
2. Christian Böhm, Ines Färber, Sergej Fries, Ulrike Korte, Johannes Merkle, Annahita Oswald, Thomas Seidl, Bianca Wackersreuther, and Peter Wackersreuther. Efficient Database Techniques for Identification with Fuzzy Vault Templates. In *Proceedings of the Special Interest Group on Biometrics and Electronic Signatures (BIOSIG)*, pages 115–126, 2011.
3. Christian Busch, Ulrike Korte, Sebastian Abt, Christian Böhm, Ines Färber, Sergej Fries, Johannes Merkle, Claudia Nickel, Alexander Nouak, Annahita Oswald, Thomas Seidl, Bianca Wackersreuther, Peter Wackersreuther, and Xuebing Zhou. Biometric Template Protection: Ein Bericht über das Projekt BioKeyS. In *Datenschutz und Datensicherheit (DuD)*, 35(3):183–191, 2011.
4. Ines Färber. Mining orthogonaler Konzepte in hochdimensionalen Datenbanken. In *Informatiktage 2009 – Fachwissenschaftlicher Informatik-Kongress*, pages 153–156, 2009.
5. 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 Clusterings. In *First Workshop on Discovering, Summarizing, and Using Multiple Clusterings (MultiClust) in conjunction with the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Minng (KDD)*, 2010.
6. Stephan Günnemann, Brigitte Boden, Ines Färber, and Thomas Seidl. Efficient Mining of Combined Subspace and Subgraph Clusters in Graphs with Feature Vectors. In *Proceedings of the 17th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD)*, pages 261–275, 2013.

7. Stephan Günnemann, Ines Färber, Brigitte Boden, and Thomas Seidl. Subspace Clustering Meets Dense Subgraph Mining: A Synthesis of Two Paradigms. In *Proceedings of the Tenth IEEE International Conference on Data Mining (ICDM)*,pages 845–850, 2010.
8. Stephan Günnemann, Ines Färber, Brigitte Boden, and Thomas Seidl. GAMer: A Synthesis of Subspace Clustering and Dense Subgraph Mining. In *Knowledge and Information Systems (KAIS)*, 40(2):243–278, 2014.
9. Stephan Günnemann, Ines Färber, Hardy Kremer, and Thomas Seidl. CoDA: Interactive Cluster Based Concept Discovery. In *Proceedings of the VLDB Endowment*, 3(2):1633–1636, 2010.
10. Stephan Günnemann, Ines Färber, Emmanuel Müller, Ira Assent, and Thomas Seidl. External Evaluation Measures for Subspace Clustering. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1363–1372, 2011.
11. Stephan Günnemann, Ines Färber, Emmanuel Müller, and Thomas Seidl. ASCLU: Alternative Subspace Clustering. In *First Workshop on Discovering, Summarizing, and Using Multiple Clusterings (MultiClust) in conjunction with the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2010.
12. Stephan Günnemann, Ines Färber, Sebastian Raubach, and Thomas Seidl. Spectral Subspace Clustering for Graphs with Feature Vectors. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM)*, pages 231–240, 2013.
13. Stephan Günnemann, Ines Färber, Matthias Rüdiger, and Thomas Seidl. SMVC: Semi-Supervised Multi-View Clustering in Subspace Projections. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, to appear, 2014.
14. Stephan Günnemann, Ines Färber, and Thomas Seidl. Multi-View Clustering Using Mixture Models in Subspace Projections. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 132–140, 2012.
15. Stephan Günnemann, Ines Färber, Kittipat Virochsiri, and Thomas Seidl. Subspace Correlation Clustering: Finding Locally Correlated Dimensions in Subspace Projections of the Data. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 352–360, 2012.

16. Stephan Günnemann, Hardy Kremer, Ines Färber, and Thomas Seidl. MC-Explorer: Interactive Exploration of Multiple (Subspace) Clustering Solutions. In *Proceedings of the IEEE International Conference on Data Mining Workshops(ICDMW)*, pages 1387–1390, 2010.
17. Stephan Günnemann, Emmanuel Müller, Ines Färber, and Thomas Seidl. Detection of Orthogonal Concepts in Subspaces of High Dimensional Data. In *Proceedings of the 18th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1317–1326, 2009.
18. Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Tutorials at the IEEE International Conference on Data Mining (ICDM)*, pages 1220, 2010.
19. Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Tutorials at the SIAM International Conference on Data Mining (SDM)*, 2011.
20. Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Tutorials at the IEEE International Conference on Data Engineering (ICDE)*, 2012.
21. Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Tutorials at the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2012.
22. Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In *Tutorials at the International Conference on Machine Learning (ICML)*, 2013.
23. Andrada Tatu, Fabian Maaß, Ines Färber, Enrico Bertini, Tobias Schreck, Thomas Seidl, and Daniel A. Keim. Subspace Search and Visualization to Make Sense of Alternative Clusterings in High-Dimensional Data. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 63–72, 2012.