2012 SIAM INTERNATIONAL CONFERENCE ON DATA MINING

3rd MultiClust Workshop:
Discovering, Summarizing and Using Multiple Clusterings

MultiClust'12

April 28, 2012

Anaheim, California, USA

Editors:
Emmanuel Müller
Karlsruhe Institute of Technology, Germany
Thomas Seidl
RWTH Aachen University, Germany
Suresh Venkatasubramanian
University of Utah, USA
Arthur Zimek
Ludwig-Maximilians-Universität München, Germany

Preface

Cluster detection is a well-established data analysis task with several decades of research. However, it also includes a large variety of different subtopics investigated by different communities such as data mining, machine learning, statistics, and database systems. "Discovering, Summarizing and Using Multiple Clusterings" is one of these emerging research fields, which is developed in all of these communities. Unfortunately, it is difficult to identify related work on this topic as different research communities are using different vocabulary and are publishing at different venues. This hinders concise and focused research as the amount of literature published every year is scattered and readers might not get the overall perspective on this topic.

The MultiClust workshop is therefore aiming at bringing together researchers that, somehow, are all tackling fundamentally identical – or rather similar – problems, around "Discovering, Summarizing and Using Multiple Clusterings". Yet they tackle these problems with different backgrounds, focus on different details, and include ideas from different research communities. This diversity is a major potential for this emerging field and should be highlighted by this workshop. In paper presentations and discussions, we would like to encourage the workshop participants to look at their own research problems from multiple perspectives.

Bridging research areas in "Multiple Clusterings" can be observed as the general idea of the entire series of MultiClust Workshops, started at ACM SIGKDD 2010, continued at ECML PKDD 2011, up to the workshop at SDM 2012. Problems known in subspace clustering met similar problems in other areas like ensemble clustering, alternative clustering, or multi-view clustering. Related research fields such as pattern mining came into the focus. Cluster exploration and visualization is another very related field, which has been discussed in all MultiClust workshops. Keeping this tradition, the 3rd MultiClust workshop, at SIAM Data Mining 2012, again encounters insights from other related fields, such as constrained clustering, distance learning, and co-learning in multiple representations.

Overall, the workshop's technical program again demonstrates the strong interest from different research communities. In particular, we have five peer-reviewed papers covering multiple research directions. They passed a competitive selection process ensuring high quality publications. Furthermore, we are pleased to have two excellent speakers giving invited talks that provide an overview on challenges in related fields: Carlotta Domeniconi (George Mason University, USA) and Shai Ben-David (University of Waterloo, Canada) contribute with their recent work in this area. And finally, in the spirit of previous workshops, the panel opens for a discussion of state-of-the-art, open challenges, and visions for future research. It wraps up the workshop by summarizing common challenges, establishing novel collaborations, and providing a guideline for topics to be addressed in following workshops.

As organizers of this workshop, we are grateful for the support of the SIAM Data Mining conference, assisting us with all organization issues. Especially we would like to thank the workshop chairs, Rui Kuang and Chandan Reddy. Furthermore, we also gratefully acknowledge the MultiClust 2012 program committee for conducting thorough reviews of the submitted technical papers. We are pleased to have some of the core researchers of the covered research topics in the MultiClust committee.

Anaheim, CA, USA, April 2012

Emmanuel Müller
Thomas Seidl
Suresh Venkatasubramanian
Arthur Zimek

Workshop Organization

Workshop Chairs

Emmanuel Müller	Karlsruhe Institute of Technology, Germany
Thomas Seidl	RWTH Aachen University, Germany
Suresh Venkatasubramanian	University of Utah, USA
Arthur Zimek	Ludwig-Maximilians-Universität München, Germany

Program Committee

Ira Assent	Aarhus University, Denmark
James Bailey	University of Melbourne, Australia
Carlotta Domeniconi	George Mason University, USA
Xiaoli Fern	Oregon State University, USA
Francesco Gullo	Yahoo! Research, Spain
Stephan Günnemann	RWTH Aachen University, Germany
Shahriar Hossain	Virginia Tech, USA
Michael Houle	National Institute of Informatics, Japan
Daniel Keim	University of Konstanz, Germany
Themis Palpanas	University of Trento, Italy
Jörg Sander	University of Alberta, Canada
Andrea Tagarelli	University of Calabria
Alexander Topchy	Nielsen Media Research, USA
Jilles Vreeken	University of Antwerp, Belgium

External Reviewers

Xuan-Hong Dang	Aarhus University, Denmark
Matthias Schäfer	University of Konstanz, Germany
Guoxian Yu	George Mason University, USA

Table of Contents

Subspace Clustering Ensembles

Carlotta Domeniconi*

Abstract

It is well known that off-the-shelf clustering methods may discover different patterns in a given set of data. This is because each clustering algorithm has its own bias resulting from the optimization of different criteria. Furthermore, there is no ground truth against which the clustering result can be validated. Thus, no cross-validation technique can be carried out to tune input parameters involved in the process. As a consequence, the user has no guidelines for choosing the proper clustering method for a given data set.

The use of *clustering ensembles* has emerged as a technique for overcoming these problems. A clustering ensemble consists of different clusterings obtained from multiple applications of any single algorithm with different initializations, or from various bootstrap samples of the available data, or from the application of different algorithms to the same data set. Clustering ensembles offer a solution to challenges inherent to clustering arising from its ill-posed nature: they can provide more robust and stable solutions by making use of the consensus across multiple clustering results, while averaging out emergent spurious structures that arise due to the various biases to which each participating algorithm is tuned, or to the variance induced by different data samples.

Another issue related to clustering is the so-called *curse of dimensionality*. Data with thousands of dimensions abound in fields and applications as diverse as bioinformatics, security and intrusion detection, and information and image retrieval. Clustering algorithms can handle data with low dimensionality, but as the dimensionality of the data increases, these algorithms tend to break down. This is because in high dimensional spaces data become extremely sparse and are far apart from each other.

A common scenario with high-dimensional data is that several clusters may exist in different subspaces comprised of different combinations of features. In many real-world problems, points in a given region of the input space may cluster along a given set of dimensions, while points located in another region may form a tight group with respect to different dimensions. Each dimension could be relevant to at least one of the clusters. Common global dimensionality reduction techniques are unable to capture such local structure of the data. Thus, a proper feature selection procedure should operate locally in the input space. Local feature selection allows one to estimate to which degree features participate in the discovery of clusters. As a result, many different *subspace* clustering methods have been proposed.

Traditionally, clustering ensembles and subspace clustering have been developed independently of one another. Clustering ensembles address the ill-posed nature of clustering, but don't address in general the curse of dimensionality problem. Subspace clustering avoids the curse of dimensionality in high-dimensional spaces, but typically requires the setting of critical input parameters whose values are unknown.

To overcome these limitations we have introduced a unified framework that is capable of handling both issues: the ill-posed nature of clustering and the curse of dimensionality. Addressing these two issues is non-trivial as it involves solving a new problem altogether: *the subspace clustering ensemble problem*. Our approach takes two different perspectives: in the one case we model the problem as a multi- and single-objective optimization one [3, 2, 1]; in the other we take a generative view, and assume that the base clusterings are generated from a hidden consensus clustering of the data [5, 4]. Both directions are promising and lead to interesting challenges. The first can yield general and efficient solutions, but requires as input the number of clusters in the consensus clustering. The second has higher complexity, but provides a principled solution to the "How many clusters?" question.

In this talk, I focus on the first approach. I introduce the formal definition of the problem of subspace clustering ensembles, and heuristics to solve it. The objective is to define methods to exploit the information provided by an ensemble of subspace clustering solutions to compute a robust consensus subspace clustering. The problem is formulated as a multi- and single-objective optimization problem where the objective functions embed both sides of the ensemble components: the data clusterings and the assignments of features to clusters.

*Department of Computer Science, George Mason University, carlotta@cs.gmu.edu

Our experimental results on real data sets demonstrate the effectiveness of the proposed methods.

References

[1] F. Gullo, C. Domeniconi, and A. Tagarelli. Projective clustering ensembles. In *Proceedings of the 2009 IEEE International Conference on Data Mining*, ICDM '09, pages 794–799. IEEE Computer Society, 2009.

[2] F. Gullo, C. Domeniconi, and A. Tagarelli. Enhancing single-objective projective clustering ensembles. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 833–838. IEEE Computer Society, 2010.

[3] F. Gullo, C. Domeniconi, and A. Tagarelli. Advancing data clustering via projective clustering ensembles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 733–744, 2011.

[4] P. Wang, C. Domeniconi, and K. B. Laskey. Latent Dirichlet Bayesian co-clustering. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, ECML PKDD '09, pages 522–537, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] P. Wang, K. B. Laskey, C. Domeniconi, and M. Jordan. Nonparametric Bayesian co-clustering ensembles. In *Proceedings of the SIAM International Conference on Data Mining*, SDM '11, pages 331–342. SIAM / Omnipress, 2011.

Cluster Center Initialization for Categorical Data Using Multiple Attribute Clustering

Shehroz S. Khan* Amir Ahmad†

Abstract

The K-modes clustering algorithm is well known for its efficiency in clustering large categorical datasets. The K-modes algorithm requires random selection of initial cluster centers (modes) as seed, which leads to the problem that the clustering results are often dependent on the choice of initial cluster centers and non-repeatable cluster structures may be obtained. In this paper, we propose an algorithm to compute fixed initial cluster centers for the K-modes clustering algorithm that exploits a multiple clustering approach that determines cluster structures from the attribute values of given attributes in a data. The algorithm is based on the experimental observations that some of the data objects do not change cluster membership irrespective of the choice of initial cluster centers and individual attributes may provide some information about the cluster structures. Most of the time, attributes with few attribute values play significant role in deciding cluster membership of individual data object. The proposed algorithm gives fixed initial cluster center (ensuring repeatable clustering results), their computation is independent of the order of presentation of the data and has log-linear worst case time complexity with respect to the data objects. We tested the proposed algorithm on various categorical datasets and compared it against random initialization and two other available methods and show that it performs better than the existing methods.

1 Introduction

Clustering aims at grouping multi-attribute data into homogenous clusters (groups). Clustering is an active research topic in pattern recognition, data mining, statistics and machine learning with diverse application such as in image analysis [19], medical applications [21] and web documentation [2].

The K-means [1] based partitional clustering methods are used for processing large numeric datasets for its simplicity and efficiency. Data mining applications require handling and exploration of heterogeneous data that contains numerical, categorical or both types of attributes together. K-means clustering algorithm fails to handle datasets with categorical attributes because it minimizes the cost function by calculating *means*. The traditional way to treat categorical attributes as numeric does not always produce meaningful results because generally categorical domains are not ordered. Several approaches have been reported for clustering categorical datasets that are based on K-means paradigm. Ralambondrainy [22] present an approach by using K-means algorithm to cluster categorical data by converting multiple category attributes into binary attributes (using 0 and 1 to represent either a category absent or present) and treat the binary attributes as numeric in the K-means algorithm. Gower and Diday [7] use a similarity coefficient and other dissimilarity measures to process data with categorical attributes. CLARA (Clustering LARge Application) [15] is a combination of a sampling procedure and the clustering program Partitioning Around Medoids (PAM). Guha et al. [8] present a robust hierarchical clustering algorithm, ROCK, that uses links to measure the similarity/proximity between a pair of data points with categorical attributes that are used to merge clusters. However this algorithm has worst case quadratic time complexity.

Huang [12] presents the K-modes clustering algorithm by introducing a new dissimilarity measure to cluster categorical data. The algorithm replaces *means* of clusters with *modes*, and use a frequency based method to update *modes* in the clustering process to minimize the cost function. The algorithm is shown to achieve convergance with linear time complexity with respect to the number of data objects. Huang [13] also pointed out that in general, the K-modes algorithm is faster than the K-means algorithm because it needs less iterations to converge.

In principle, K-modes clustering algorithm functions similar to K-means clustering algorithm except for the cost function it minimizes, and hence suffers from the same drawbacks. Likewise K-means, the K-modes clustering algorithm assumes that the number of clusters, K, is known in advance. Fixed number of K clus-

*University of Waterloo, Ontario, Canada.
†King Abdulaziz University, Rabigh, Saudi Arabia.

ters can make it difficult to predict the actual number of clusters in the data that may mislead the interpretations of the results. It also fall into problems when clusters are of differing sizes, density and non-globular shapes. K-means does not guarantee unique clustering due to random choice of initial cluster centers that may yield different groupings for different runs [14]. Similarly, K-modes algorithm is also very sensitive to the choice of initial centers, an improper choice may yield highly undesirable cluster structures. Random initialization is widely used as a seed for K-modes algorithm due to its simplicity, however, this may lead to undesirable and/or non-repeatable clustering results. Machine learning practioners find it difficult to rely on the results thus obtained and several re-runs of K-modes algorithm may be required to arrive at a meaningful conclusion.

There are several attempts to initialize cluster centers for K-modes algorithm, however, most of these methods suffer from either one or more of the three drawbacks: a) the initial cluster center computation methods are non-linear in time complexity with respect to the number of data objects b) the initial modes are not fixed and possess some kind of randomness in the computation steps and c) the methods are dependent on the presentation of order of data objects (details are discussed in Section 2). In this paper, we present a multiple clustering approach that infers cluster structure information from the attributes using their attribute values present in the data for computing initial cluster centers. Our proposed algorithm performs mulitple partitional clustering on different attributes of the data to generate fixed initial centers (modes), is independent of the order of presentation of data and thus gives fixed clustering results. The proposed algorithm has worst case log-linear time complexity with respect to the number of data objects.

The rest of the paper is organized as follows. In Section 2 we review research work on cluster center initialization for K-modes algorithm. Section 3 briefly discusses the K-modes clustering algorithm. In Section 4 we present the proposed approach to compute initial modes using multiple clustering that takes contributions from different attribute values of individual attributes to determine distinguishable clusters in the data. In Section 5, we present the experimental analysis of the proposed method on various categorical datasets to compute initial cluster centers, compare it with other methods and show improved and consistent clustering results. Section 6 concludes the presentation with pointers to future work.

2 Related Work

The K-modes algorithm [12] extends the K-means paradigm to cluster categorical data and requires random selection of initial center or modes. The random initialization of cluster center may only work well when one or more chosen initial centers are close to actual centers present in the data. In the most trivial case, the K-modes algorithm keeps no control over the choice of initial centers and therefore repeatability of clustering results is difficult to achieve. Moreover, inappropriate choice of initial cluster centers can lead to undesirable clustering results. Hence, it is desirable to start K-modes clustering with fixed initial centers that resemble the true representatives centers of the clusters. Below we provide a short review of the research work done to compute initial cluster centers for K-modes clustering algorithm and discuss their associated problems.

Huang [13] propose two approaches for initializing the clusters for K-modes algorithm. In the first method, the first K distinct data objects are chosen as initial K-modes, whereas the second method calculates the frequencies of all categories for all attributes and assign the most frequent categories equally to the initial K-modes. The first method may only work if the top K data objects come from disjoint K clusters, therefore it is dependent on order of presentation of data. The second method is aimed at choosing diverse cluster center that may improve clustering results, however a uniform criteria for selecting K-initial centers is not provided. Sun Yin et al. [23] present an experimental study on applying Bradley et al.'s iterative initial-point refinement algorithm [3] to the K-modes clustering to improve the accuracy and repetitiveness of the clustering results. Their experiments show that the K-modes clustering algorithm using refined initial points leads to higher precision results much more reliably than the random selection method without refinement. This method is dependent on the number of cases with refinements and the accuracy value varies. Khan and Ahmad [16] use Density-based Multiscale Data Condensation [20] approach with Hamming distance to extract K initial points, however, their method has quadratic complexity with respect to the number of data objects. He [10] presents two farthest point heuristic for computing initial cluster centers for K-modes algorithm. The first heuristic is equivalent to random selection of initial cluster centers and the second uses a deterministic method based on a scoring function that sums the frequency count of attribute values of all data objects. This heuristic does not explain how to choose a point when several data objects have same scores, and if it randomly break ties, then fixed centers cannot be guaranteed. Wu et al. [24] develop a density based method to compute

the K initial modes which has quadratic complexity. To reduce its complexity to linear they randomly select square root of the total points as a sub-sample of the data, however, this step introduces randomness in the final results and repeatability of clustering results may not be achieved. Cao et al. [4] present an initialization method that consider distance between objects and the density of the objects. A major drawback of this method is that it has quadratic complexity. Khan and Kant [18] propose a method that is based on the idea of evidence accumulation for combining the results of multiple clusterings [6] and only focus on those data objects that are more less vulnerable to the choice of random selection of modes and to choose the most diverse set of modes among them. Their experiment suggest that the initial modes outperform the random choice, however the method does not guarantee fixed choice of initial modes.

In the next section, we briefly describe the K-modes clustering algorithm.

3 K-Modes Algorithm for Clustering Categorical Data

Due to the limitation of the dissimilarity measure used by traditional K-means algorithm, it cannot be used to cluster categorical dataset. The K-modes clustering algorithm is based on K-means paradigm, but removes the numeric data limitation whilst preserving its efficiency. The K-modes algorithm extends the K-means paradigm to cluster categorical data by removing the barrier imposed by K-means through following modifications:

1. Using a simple matching dissimilarity measure or the Hamming distance for categorical data objects
2. Replacing means of clusters by their modes (cluster centers)

The simple matching dissimilarity measure can be defined as following. Let X and Y be two categorical data objects described by m categorical attributes. The dissimilarity measure $d(X,Y)$ between X and Y can be defined by the total mismatches of the corresponding attribute categories of two objects. Smaller the number of mismatches, more similar the two objects are. Mathematically, we can say

$$d(X,Y) = \sum_{j=1}^{m} \delta(x_j, y_j) \tag{3.1}$$

where $\delta(x_, = y_j) = \begin{cases} 0 & (x_j = y_j) \\ 1 & (x_j \neq y_j) \end{cases}$

$d(X,Y)$ gives equal importance to each category of an attribute.

Let Z be a set of categorical data objects described by categorical attributes, $A_1, A_2, \ldots A_m$. When the above is used as the dissimilarity measure for categorical data objects, the cost function becomes

$$C(Q) = \sum_{i=1}^{n} d(Z_i, Q_i) \tag{3.2}$$

where Z_i is the i^{th} element and Q_i is the nearest cluster center of Z_i. The K-modes algorithm minimizes the cost function defined in Equation 3.2.

The K-modes assumes that the knowledge of number of natural grouping of data (i.e. K) is available and consists of the following steps: -

1. Create K clusters by randomly choosing data objects and select K initial cluster centers, one for each of the cluster.
2. Allocate data objects to the cluster whose cluster center is nearest to it according to equation 3.2.
3. Update the K clusters based on allocation of data objects and compute K new modes of all clusters.
4. Repeat step 2 to 3 until no data object has changed cluster membership or any other predefined criterion is fulfilled.

4 Multiple Attribute Clustering Approach for Computing Initial Cluster Centers

Khan and Ahmad [17] show that for partitional clustering algorithms, such as K-Means, some of the data objects are very similar to each other and that is why they share same cluster membership irrespective to the choice of initial cluster centers. Also, an individual attribute may provide some information about initial cluster center. He et al. [11] present a unified view on categorical data clustering and cluster ensemble for the creation of new clustering algorithms for categorical data. Their intuition is that attributes present in a categorical data contributes to the final cluster structure. They consider the attribute values of an attribute as cluster labels giving "best clustering" without considering other attributes and created a cluster ensemble. We take motivation from these research works and propose a new cluster initialization algorithm for categorical datasets that perform multiple clustering on different attributes and uses distinct attribute values as cluster labels as a cue to find consistent cluster structure and an aid in computing better initial centers. The proposed approach is based on the following experimental observations (assuming that the desired number of clusters, K, are known):

1. Some of the data objects are very similar to each other and that is why they have same cluster membership irrespective of the choice of initial cluster centers [17].

2. There may be some attributes in the dataset whose number of attribute values are less than or equal to K. Due to fewer attribute values per cluster, these attributes shall have higher discriminatory power and will play a significant role in deciding the initial modes as well as the cluster structures. We call them as *Prominent Attributes (P)* .

3. There may be few attributes whose number of attribute values are greater than K. The many atrribute values in these attributes will be spread out per cluster, add little to determine proper cluster structure and contribute less in deciding the initial representative modes of the clusters.

The main idea of the proposed algorithm is to partition the data, for every prominent attribute based on its attribute values, and generate a *cluster string* that contains the respective cluster allotment labels of the full data. This process yields a number of cluster strings that represent different partition views of the data. As noted above, some data objects will not be affected by choosing different cluster centers and their cluster strings will remain same. The algorithm assumes that the knowledge of natural clusters in the data i.e. K is available and merges similar cluster strings into K partitions. This step will group similar cluster strings into K clusters. In the final step, the cluster strings within each K clusters are replaced by the corresponding data objects and modes of every K cluster is computed that serves as the initial centers for the K-modes algorithm. The algorithmic steps of the proposed approach are presented below.

Algorithm: Compute Initial Modes. Let Z be a categorical dataset with N data objects embedded in M dimensional feature space.

1. Calculate the number of Prominent Attributes ($\#P$)

2. If $\#P > 0$, then use these Prominent Attributes for computing initial modes by calling *getInitialModes(Attributes P)*

3. If $\#P = 0$ i.e. there are no Prominent Attributes in the data, or if $\#P = M$ i.e. all attributes are Prominent Attributes, then use all attributes and call *getInitialModes(Attributes M)*

Algorithm: *getInitialModes(Attributes A)*

1. For every $i \in A$, i=1,2...A, repeat step 2 to 4. Let j denotes the number of attribute values of i^{th} attribute. Note that if A is P then $j \leq K$, else if A is M then $j > K$.

2. Divide the dataset into j clusters on the basis of these j attribute values such that data objects with different values (of this attribute i) fall into different clusters.

3. Compute j M-dimensional modes, and partition the data by performing K-modes clustering that consumes them as initial modes.

4. Assign cluster label to every data object as S_{ti}, where t=1,2...N

5. Generate cluster string G_t corresponding to every data object by storing the cluster labels from S_{ti}. Every data object will have A class labels.

6. Find distinct cluster strings from G_t, count their frequency, and sort them in descending order. Their count, K', is the number of distinguishable clusters.

   There arise three possibilities:

   (a) $K' > K$ – Merge similar distinct cluster string of G_t into K clusters and compute initial modes (details presented in Section 4.1)

   (b) $K' = K$ – Distinct cluster strings of G_t matches the desired number of clusters in the data. Glean the data objects corresponding to these K cluster strings, which will serve as initial modes for the K-modes clustering algorithm.

   (c) $K' < K$ – An obscure case may arise where the number of distinct cluster strings are less than the chosen K (assumed to represent the natural clusters in the data). This case will only happen when the partitions created based on the attribute values of A attributes groups the data in the same clusters every time. A possible scenario is when the attribute values of all attributes follow almost same distribution, which is normally not the case in real data. This case also suggests that probably the chosen K does not resemble with the natural grouping and it should be changed to a lesser value. The role of attributes with attribute values greater than K has to be investigated in this case. This particular case is out of the scope of the present paper.

4.1 Merging Clusters As discussed in step 6 of algorithm *getInitialModes(Attributes A)*, there may arise a case when $K' > K$, which means that the number of distinguishable clusters obtained by the algorithm are more than the desired number of clusters in the data. Therefore, K' clusters must be merged to arrive at K clusters. As these K' clusters represent distinguishable clusters, a trivial approach could be to sort them in order of cluster string frequency and pick the top K cluster strings. A problem with this method is that it cannot be ensured that the top K most frequent cluster strings are representative of K clusters. If more than one cluster string comes from same cluster then the K-modes algorithm will fall apart and will give undesirable clustering results. This observational fact is also verified experimentally and holds to be true.

Keeping this issue in mind, we propose to use the hierarchical clustering method [9] to merge K' distinct cluster strings into K clusters. Hierarchical clustering has the disadvantage of having quadratic time complexity with respect to the number of data objects. In general, K' cluster strings will be less than N. However, to avoid extreme case such as when $K' \approx N$, we only choose the most frequent $N^{0.5}$ distinct cluster strings of G_t. This will make the hierarchical algorithm log-linear with the number of data objects (K' or $N^{0.5}$ distinct cluster strings here). The infrequent cluster strings can be considered as outliers or boundary cases and their exclusion does not affect the computation of initial modes. In the best case, when $K' \ll N^{0.5}$, the time complexity effect of log-linear hierarchical clustering will be minimal. The hierarchical clusterer merges K' ($N^{0.5}$ in worst case) distinct cluster strings of G_t by labelling them in the range of $1 \dots K$. For every cluster label $k = 1 \dots K$, group the data objects corresponding to the cluster string with label k and compute the group modes. This process generates K M-dimensional modes that are to be used as initial modes for K-modes clustering algorithm.

4.2 Choice of Attributes. The proposed algorithm starts with the assumption that there exists prominent attributes in the data that can help in obtaining distinguishable cluster structures that can be merged to obtain initial cluster centers. In the absence of any prominent attributes (or if all attributes are prominent), *Vanilla Approach*, all the attributes are selected to find initial modes. Since attributes other can prominent attributes contain attribute values more than K, a possible repercussion is the increased number of distinct cluster strings G_t due to the availability of more cluster allotment labels. This implies an overall reduction in the individual count of distinct cluster strings and many small clusters may arise side-by-side. Since the hierarchical clusterer imposes a limit of $\sqrt{N}$ on the top cluster strings to be merged, few distinguishable cluster could lay outside the bound during merging. This may lead to some loss of information and affects the quality of the computed initial cluster centers. The best case occurs when the number of distinct cluster strings is less than or equal to $\sqrt{N}$.

4.3 Evaluating Time Complexity The above proposed algorithm to compute initial cluster centers has two parts, namely, *getInitialModes(Attributes A)* and merging of clusters. In the first part, the K-modes algorithm is run P times (in the worst case M times). As the K-modes algorithm is linear with respect to the size of the dataset [12], the worst case time complexity will be $M.O(rKMN)$, where r is the number of iterations needed for convergence and $\ll$ N. In the second part of the algorithm, the hierarchical clustering is used. The worst case complexity of the hierarchical clustering is $O(N^2 logN)$. As the proposed approach chooses distinct cluster strings that are less than or equal to $N^{0.5}$, the worst case complexity becomes $O(NlogN)$. Combining both the parts, the worst case time complexity of the proposed approach becomes $(M.O(rKMN) + O(NlogN))$, which is log-linear with respect to the size of the dataset.

5 Experimental Analysis

5.1 Datasets. To evaluate the performance of the proposed initialization method, we use several pure categorical datasets from the UCI Machine Learning Repository [5]. All the datasets have multiple attributes and varied number of classes, and some of the dataset contain missing values. A short description for each dataset is given below.

Soybean Small. The soybean disease dataset consists of 47 cases of soybean disease each characterized by 35 multi-valued categorical variables. These cases are drawn from four populations, each one of them representing one of the following soybean diseases: D1-Diaporthe stem canker, D2-Charcoat rot, D3-Rhizoctonia root rot and D4-Phytophthorat rot. Ideally, a clustering algorithm should partition these given cases into four groups (clusters) corresponding to the diseases. The clustering results on soybean data are shown in Table 2.

Breast Cancer Data. This data has 699 instances with 9 attributes. Each data object is labeled as benign (458 or 65.5%) or malignant (241 or 34.5%). There are 9 instances in Attribute 6 and 9 that contain a missing

(i.e. unavailable) attribute value. The clustering results of breast cancer data are shown in Table 3.

Zoo Data. It has 101 instances described by 17 attributes and distributed into 7 categories. All of the characteristics attributes are Boolean except for the character attribute corresponds to the number of legs that lies in the set 0, 2, 4, 5, 6, 8. The clustering results of zoo data are shown in Table 4.

Lung Cancer Data. This dataset contains 32 instances described by 56 attributes distributed over 3 classes with missing values in attributes 5 and 39. The clustering results for lung cancer data are shown in Table 5.

Mushroom Data. Mushroom dataset consists of 8124 data objects described by 22 categorical attributes distributed over 2 classes. The two classes are edible (4208 objects) and poisonous (3916 objects). It has missing values in attribute 11. The clustering results for mushroom data are shown in Table 6.

5.2 Comparison and Performance Evaluation Metric. We compared the proposed cluster initial center against the random initialization method and the methods described by Cao et al. [4] and Wu et al. [24]. For random initialization, we randomly group data objects into K clusters and compute their modes to be used as initial centers. The reported results are an average of 50 such runs.

To evaluate the performance of clustering algorithms and for fair comparison of results, we used the performance metrics used by Wu et al [24] that are derived from information retrieval. If a dataset contains K classes for any given clustering method, let a_i be the number of data objects that are correctly assigned to class C_i, let b_i be the number of data objects that are incorrectly assigned to class C_i, and let c_i be the data objects that are incorrectly rejected from class C_i, then precision, recall and accuracy are defined as follows:

$$PR = \frac{\sum_{i=1}^{K} \left(\frac{a_i}{a_i + b_i} \right)}{K} \tag{5.3}$$

$$RE = \frac{\sum_{i=1}^{K} \left(\frac{a_i}{a_i + c_i} \right)}{K} \tag{5.4}$$

$$AC = \frac{\sum_{i=1}^{K} a_i}{N} \tag{5.5}$$

Table 1: Effect of choosing different number of attributes.

~~Dataset~~	Proposed		Vanilla		$\sqrt{N}$
	#P	#CS	#A	#CS	
Soybean	20	21	35	25	7
Zoo	16	7	17	100	11
Breast-Cancer	9	355	9	355	27
Lung-Cancer	54	32	56	32	6
Mushroom	5	16	22	683	91

5.3 Effect of Number of Attributes. To test the intuition discussed in Section 4.2, we performed a comparative analysis on the effect of number of selected attributes on the number of distinct cluster strings. In Table 1, #P is the number of prominent attributes, #A is the total number of attributes in the data, #CS is the number of distinct cluster string and $\sqrt{N}$ is the limit on the number of top cluster strings to be merged using hierarchical clustering. The table shows that choosing a *Vanilla* approach (all attributes) leads to higher #CS, whereas with the proposed approach the number of distinct cluster strings are much lesser. For breast cancer data, all the attributes were prominent therefore #P and #A are same and hence same #CS. For lung cancer data #P $\approx$ #A therefore #CS are same. Due to the limit of merging top $\sqrt{N}$ cluster string (and reasons described in Section 4.2), clustering results using a *Vanilla* approach is worse than the proposed approach and are not reported in the paper. It is to be noted that the #CS using proposed approach for Zoo and Mushroom data are within the bounds of $\sqrt{N}$ limit.

5.4 Clustering Results. It can be seen from Table 2 to 6 that the proposed initialization method outperforms random cluster initialization for categorical data in accuracy, precision and recall. Another advantage of the proposed method is that it generates fixed cluster centers, whereas the random initialization method does not. Therefore, repeatable and better cluster structures can be obtained using the proposed method. In comparison to the initialization methods of Cao et al. and and Wu et al., the findings can be summarized as:

- in terms of accuracy, the proposed method outperforms or equals other methods in 4 cases and perform worse in one case.
- in terms of precision, the proposed method performs well or equals other methods in 2 cases while performs worse in 3 cases.
- in terms of recall, the proposed method outperforms or equals other methods in 4 cases whereas

it perform worse in 1 case.

Table 2: Clustering results for Soybean data

	Random	**Wu**	**Cao**	**Proposed**
AC	0.8644	**1**	**1**	0.9574
PR	0.8999	**1**	**1**	0.9583
RE	0.8342	**1**	**1**	0.9705

Table 3: Clustering results for Breast Cancer data

	Random	**Wu**	**Cao**	**Proposed**
AC	0.8364	0.9113	0.9113	**0.9127**
PR	0.8699	**0.9292**	**0.9292**	**0.9292**
RE	0.7743	0.8773	0.8773	**0.8783**

Table 4: Clustering results for Zoo data

	Random	**Wu**	**Cao**	**Proposed**
AC	0.8356	0.8812	0.8812	**0.891**
PR	0.8072	**0.8702**	**0.8702**	0.7302
RE	0.6012	0.6714	0.6714	**0.8001**

Table 5: Clustering results for Lung Cancer data

	Random	**Wu**	**Cao**	**Proposed**
AC	**0.5210**	0.5	0.5	0.5
PR	0.5766	0.5584	0.5584	**0.6444**
RE	0.5123	0.5014	0.5014	**0.5168**

Table 6: Clustering results for Mushroom data

	Random	**Wu**	**Cao**	**Proposed**
AC	0.7231	0.8754	0.8754	**0.8815**
PR	0.7614	**0.9019**	**0.9019**	0.8975
RE	0.7174	0.8709	0.8709	**0.8780**

The above results are very encouraging due to the fact that the worst case time complexity of the proposed method is log-linear, whereas the method of Cao et al. has quadratic complexity and the method of Wu et al. induces random selection of data points. The accuracy values of proposed method are mostly better than or equal to other methods, which implies that the proposed approach is able to find fixed initial centers that are close to the actual centers of the data. The only case where the proposed method perform worse in all three performance metric is the soybean dataset. We observe that on some datasets the proposed method gives worse values for precision, which implies that in those cases some data objects from non-classes are getting clustered in given classes. The recall values of proposed method are mostly better than the other methods, which suggests that the proposed approach tightly controls the data objects from given classes to be not clustered to non-classes. Breast cancer data has no prominent attribute in the data and uses all the attributes and produces comparable results to other methods. Lung cancer data, though smaller in size has high dimension and the proposed method is able to produce better precision and recall rates than other methods. It is also observed that the proposed method perform well on large dataset such as mushroom data with more than 8000 data objects. In our experiment, we did not get a scenario where the distinct cluster strings are less than the desired number of clusters. The proposed algorithm is also independent of the order of presentation of data due to he way mode is computed for different attributes.

6 Conclusions

The results attained by the K-modes algorithm for clustering categorical data depends intrinsically on the choice of random initial cluster center, that can cause non-repeatable clustering results and produce improper cluster structures. In this paper, we propose an algorithm to compute initial cluster center for categorical data by performing multiple clustering on attribute values of attributes present in the data. The present algorithm is developed based on the experimental fact that similar data objects form the core of the clusters and are not affected by the selection of initial cluster centers, and that individual attribute also provide useful information in generating cluster structures, that eventually leads to computing initial centers. In the first pass, the algorithm produces distinct distinguishable clusters, that may be greater than, equal to or less than the desired number of clusters (K). If it is greater than K then hierarchical clustering is used to merge similar cluster strings into K clusters, if it is equal to K then data objects corresponding to cluster strings can be directly used as initial cluster centers. An obscure possibility arises when cluster strings are less than K, in which case either the value of K is to be reduced, or assumed that the current value of K is not true representative of the desired number of clusters. However, in our experiment we did not get such situation, largely because it can happen in a rare occurence when all the attribute values of different attributes cluster the data in the same way. These initial cluster centers when used as seed to K-modes clustering algorithm, improves the accuracy of the traditional K-modes clustering algorithm that uses random modes as starting point. Since there is a def-

inite choice of initial modes (zero standard deviation), consistent and repetitive clustering results can be generated. The proposed method also does not depend on the way data is ordered. The performance of the proposed method is better than or equal to the other two methods on all datasets except one case. The biggest advantage of the proposed method is the worst case log-linear time complexity of computation and fixed choice of initial cluster centers, whereas both the other two methods lack either one of them.

In scenarios when the desired number of clusters are not available at hand, we would like to extend the proposed multi-clustering approach for categorical data for finding out the natural number of clusters present in the data in addition to computing the initial cluster centers for such case.

References

[1] Michael R. Anderberg. *Cluster analysis for applications*. Academic Press, New York, 1973.

[2] Daniel Boley, Maria Gini, Robert Gross, Eui-Hong Han, George Karypis, Vipin Kumar, Bamshad Mobasher, Jerome Moore, and Kyle Hastings. Partitioning-based clustering for web document categorization. *Decision Support Systems.*, 27:329–341, December 1999.

[3] Paul S. Bradley and Usama M. Fayyad. Refining initial points for k-means clustering. In Jude W. Shavlik, editor, *ICML*, pages 91–99. Morgan Kaufmann, 1998.

[4] Fuyuan Cao, Jiye Liang, and Liang Bai. A new initialization method for categorical data clustering. *Expert Systems and Applications*, 36:10223–10228, 2009.

[5] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[6] Ana L. N. Fred and Anil K. Jain. Data clustering using evidence accumulation. In *ICPR (4)*, pages 276–280, 2002.

[7] K. Chidananda Gowda and E. Diday. Symbolic clustering using a new dissimilarity measure. *Pattern Recogn.*, 24:567–578, April 1991.

[8] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. In *Proceedings of the 15th International Conference on Data Engineering, 23-26 March 1999, Sydney, Austrialia*, pages 512–521. IEEE Computer Society, 1999.

[9] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. In *in SIGKDD Explorations*, volume 11 of *1*, 2009.

[10] Zengyou He. Farthest-point heuristic based initialization methods for k-modes clustering. *CoRR*, abs/cs/0610043, 2006.

[11] Zengyou He, Xiaofei Xu, and Shengchun Deng. A cluster ensemble method for clustering categorical data. *Information Fusion*, 6(2):143–151, 2005.

[12] Zhexue Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Research Issues on Data Mining and Knowledge Discovery*, 1997.

[13] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.*, 2(3):283–304, 1998.

[14] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[15] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990.

[16] Shehroz S. Khan and Amir Ahmad. Computing initial points using density based multiscale data condensation for clustering categorical data. In *Proc. of 2nd Int'l Conf. on Applied Artificial Intelligence*, 2003.

[17] Shehroz S. Khan and Amir Ahmad. Cluster center initialization algorithm for k-means clustering. *Pattern Recognition Letters*, 25:1293–1302, 2004.

[18] Shehroz S. Khan and Shri Kant. Computation of initial modes for k-modes clustering algorithm using evidence accumulation. In *Proceedings of the 20th international joint conference on Artificial intelligence (IJCAI)*, pages 2784–2789, 2007.

[19] Jiri Matas and Josef Kittler. Spatial and feature space clustering: Applications in image analysis. In *CAIP*, pages 162–173, 1995.

[20] Pabitra Mitra, C. A. Murthy, and Sankar K. Pal. Density-based multiscale data condensation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):734–747, 2002.

[21] Euripides G. M. Petrakis and Christos Faloutsos. Similarity searching in medical image databases. *IEEE Transactions on Knowledge Data Engineering.*, 9(3):435–447, 1997.

[22] H. Ralambondrainy. A conceptual version of the k-means algorithm. *Pattern Recognition Letters*, 16(11):1147–1157, 1995.

[23] Ying Sun, Qiuming Zhu, and Zhengxin Chen. An iterative initial-points refinement algorithm for categorical data clustering. *Pattern Recognition Letters*, 23(7):875–884, 2002.

[24] Shu Wu, Qingshan Jiang, and Joshua Zhexue Huang. A new initialization method for clustering categorical data. In *Proceedings of the 11th Pacific-Asia conference on Advances in knowledge discovery and data mining*, PAKDD'07, pages 972–980, Berlin, Heidelberg, 2007. Springer-Verlag.

Co-RCA: Unsupervised Distance-Learning for Multi-View Clustering

Matthias Schubert Hans-Peter Kriegel
Institute for Computer Science Ludwig Maximilians University Munich

Oettingenstr. 67, D-80538 Munich
{ schubert, kriegel}@dbs.ifi.lmu.de

Abstract

There has been a considerable effort on tuning distance metrics for supervised problems such as kNN classification. Most methods in metric learning aim at deriving an optimized matrix for the Mahanalobis distance. However, these methods cannot be applied for clustering methods because there is no training set indicating which instances are similar and which are not. In this paper, we will show that the lack of training data can be overcome by using multiple views. Our new method Co-RCA combines the idea of co-learning with relevant component analysis (RCA). Based on the assumption that the closest pairs in any useful feature space are semantically similar as well, it is possible to improve distance measures in unsupervised learning tasks such as clustering. Our experiments demonstrate that Co-RCA can improve the semantic meaning of the distances for two image data sets.

1 Introduction

One of the most essential aspects in developing successful knowledge discovery processes is finding suitable features spaces. In a well-suited feature space the distance between two objects should reflect the semantic similarity holding in the given application. There are basically two methods of optimizing a feature space to better represent the underlying similarity structure. The first method is to directly manipulate the feature space, for example removing redundant and unimportant features or deriving new features. The second method is manipulating the employed comparison function which is used to compute object similarity. In most cases, the comparison function is either a kernel function or a distance metric. The most common framework for optimizing features spaces are affine transformations, like principal component analysis (PCA) and its extensions. The most common framework for metric learning is the Mahalanobis distance or quadratic form which is based on a matrix modeling how the features are correlated. It can be shown that any matrix that guarantees that the Mahalanobis distance is a metric corresponds to a linear basis transformation where the Euclidian distance in the transformed space is equivalent the Mahalanobis distance in the original space. There exists a wide variety of methods for learning a proper Mahalanobis distance or the equivalent transformation matrix for supervised problems [18]. The core idea of any such method is to decrease the distances between similar objects while increasing the distances between dissimilar objects. To construct an optimization problem, metric learning methods employ examples to measure the contrast between the similarity being computed in the features space and the similarity being observed by a human expert. For classification, information about similarity is usually drawn from the class labels. Objects belonging to the same class are considered to be similar and objects belonging to different classes are considered to be dissimilar. Requiring that all objects belonging to the same class are closer to each other than to any object of any different class is often a too strict goal. Therefore, many methods relax this requirement. For example, it might be sufficient to optimize the similarity to the k-closest neighbors from the same class instead of the similarity to all other members of the same class. Though there are many successful methods for metric learning for supervised tasks, optimizing feature spaces without examples for similar and dissimilar objects received less attention by researchers. Though there are many methods which try to reduce the dimensionality of feature spaces such as PCA, the positive effect on the quality of the feature space is mainly achieved by removing redundant information and rescaling the remaining features. In some cases, the observed variance of the resulting feature values gives hints on the importance of a feature. However, a large variance can be caused by a feature which is generally measured by a larger scale. Thus, the feature does not need to have an increased importance. To conclude, though methods like PCA yield a powerful tool for improving data quality, the available information is often not enough to

improve the meaning of the observed distances.

In this paper, we consider the case of data which is represented in multiple feature spaces also called multi-view or multi-represented data. The core idea of our approach is to improve the similarity information in a feature space by employing the similarity information drawn from other views. Thus, the method is unsupervised but integrates information about object similarity which is not derived from the particular feature space that is currently optimized. Technically, our approach is based on the idea of ensemble learning and co-training in particular. Therefore, there are two important requirements to the underlying feature spaces. The first is that the information about object similarity provided by different views should be sufficiently independent from each other. For example, using texture and color features should yield independent feature spaces because each of the feature spaces is based on a different characteristic of a pixel image. The second requirement of ensemble learning is that each weak learner is sufficiently accurate. In our setting it is required that there must be a sufficiently large number of cases where a small distance corresponds to a large similarity and large distance corresponds to a small degree of similarity. Our algorithm Co-RCA is derived from the co-training framework introduced in [4]. In a first step, we derive examples indicating similarity or dissimilarity for a smaller number of object comparisons from an initial feature representation. In particular, we consider the k-closest pairs and k-farthest pairs of objects and use them to determine similar and dissimilar object pairs. Given these examples, we employ relevant component analysis(RCA) to improve the structure of the next feature space. Thus, the set of examples for similar and dissimilar objects is increased with each iteration. The algorithm terminates when no new example pairs for similar objects are retrieved from either feature space. Thus, the k-closest pair in all views are stable under the current transformation.

The rest of the paper is organized as follows: In section 2, we review related work on metric learning and feature reduction. Section 3 will shortly survey the two techniques our method is based on, co-training and RCA. The new algorithm Co-RCA is introduced in section 4. The results of our experimental evaluation are described in section 5. The paper concludes with a brief summary in section 6.

2 Related Work

In this section, we briefly review existing approaches to metric learning and affine optimizations of vector spaces.

Most distance learning try to optimize the Mahalanobis distance. In the following, we give a short summary of existing metric learning approaches. A detailed survey can be found in [18]. Among supervised approaches one of the first methods is Fisher's Linear Discriminant (FLD) [10]. It maximizes the ratio of the between-class variance and the within-class variance using a generalized eigenvalue decomposition. This method has been extended by Belhumeur et al. [2] to the Fisherfaces approach. It precedes FLD with a reduction of the input space to its principal components and can thus filter unreliable input dimensions. With RCA [1], Bar-Hillel et al. focus on the problem of minimizing within-*chunklet* variance. They argue that between-class differences are less informative than within-class differences and that class assignments frequently occur in such a way that only pairs of equally-labelled objects can be extracted. These pairs are extended into chunklets (sets) of equivalent objects. The inverse chunklet covariance matrix is used for calculating the Mahalanobis distance. NCA [11] proposed by Goldberger et al. optimizes an objective function based on a soft neighborhood assignment evaluated via the leave-one-out error. This setting makes it more resistant against multi-modal distributions. The loss function is differentiated and then optimized by general gradient descent approaches like delta-bar-delta or conjugate gradients. The result of this optimization is a Mahalanobis distance directly aimed at improving nearest-neighbor classification. The objective function is, however, not guaranteed to be convex. With Information-Theoretic Metric Learning (ITML) [8], Davis et al. propose a low-rank kernel learning problem which generates a Mahalanobis matrix subject to an upper bound for inner-class distances and a lower bound to between-class distances. They regularize by choosing the matrix closest to the identity matrix and introduce a way to reduce the rank of the learning problem. LMNN (Large Margin Nearest Neighbor) [16] by Weinberger et al. is based on a semi-definite program for directly learning a Mahalanobis matrix M. They require *k-target neighbors* for each input object x, specifying a list of objects, usually of the same class as x, which should always be mapped closer to x than any object belonging to any other class. These k-target neighbors are the within-class k-nearest neighbors. Hence, the loss function consists of two terms for all data points x. One penalizing the distance of x to its k-target neighbors and a second penalizing close objects being closer to x than any of its target neighbors.LMNN requires a specialized solver in order to be run on larger data sets. For multi-view metric learning, there are already some methods for semi-supervised settings [17, 19]. Though these methods do exploit multiple views as Co-RCA does, they still re-

quire label information even if the amount of labels is expected to be limited.

The main idea of unsupervised approaches is to reduce the feature space to a lower-dimensional space in order to eliminate noise and enable a more efficient object comparison. The criteria for selecting such a subspace are manifold. Principal Component Analysis (PCA) [12], builds an orthogonal basis aimed at best preserving the data's variance, Multidimensional Scaling (MDS) [7] seeks the transformation which best preserves the geodesic distances and Independent Component Analysis (ICA) [6] targets a subspace that guarantees maximal statistical independence. ISOMAP [15] by Tenenbaum et al. is a non-linear enhancement of the MDS principle, in identifying the geodesic manifold of the data and preserving its intrinsic geometry. Other unsupervised approaches (e.g. [14, 3]) try to fulfill the above criteria on a local scale.

3 Preliminaries

Co-Training The co-training framework was introduced by Blum and Mitchell and in [4]. In particular their original work aimed at the improvement of webpage classification under the condition of a limited set of labeled training data. The original set of experiments used two views on webpages. The first view is text vectors being derived from the actual content of the webpage. The second view consists of the anchor texts of the hyper links being directed at the webpage. Let us note that both views are sufficiently independent because the words in the anchor text need not appear in the actual content as well. When starting the algorithm there is a large set of web pages being described by both views. However, only a limited portion of the pages are labeled w.r.t. the given classification task. The algorithm starts with training a text classifier on the first view based on the already labeled pages. Let us note that the resulting classifier must meet a sufficiently large accuracy to act as a weak classifier because co-training is in its essence an ensemble learning approach. Given the first classifier the method samples a set of unlabeled webpages and uses the classifier to predict class labels for these pages. In the next step, the algorithm joins the newly labeled objects with the already labeled objects into a new training set and switches the views. Thus, in the next step a second classifier is trained based on the extended set of training objects. Since the second classifier is based on a different view of the data, it models the classes independently even though its training set is partly made of automatically labeled example vectors. Again the classifier is used to extend the training set. Afterwards the algorithm switches to the next view and proceeds in the same way. The algorithm is stopped when a predefined number of iterations is computed.

Though the general process of Co-RCA and co-training have a similar algorithmic scheme, there are considerable differences. First of all, co-training is aimed at the supervised task of classification with limited label information. Co-RCA offers a method to improve the expressiveness w.r.t. multiple views without any task specific labeling. Thus, Co-RCA does not require any labels but is leveraged by the assumption that the closest and most distant object pairs in a view are really similar or dissimilar respectively.

Fisher Faces and RCA Fisher Faces [2] are an extension of the well known linear classifier known as Fisher's discriminant analysis (FDA). The core idea of FDA is to compare feature correlations occurring between objects of the same class to those occurring between objects belonging to different classes. Thus, the method needs to derive two covariance matrices: The within class matrix M_w and the between class matrix M_b. To determine M_w a covariance matrix for each class c M_c is build and afterwards summed up over all classes C : $M_w = \sum_{c \in C} M_c$

M_b is computed as the covariance matrix of the class means μ_c for each class $c \in C$.

The idea of FDA is now to find dimensions that maximize the ratio of the covariance between classes while considering the covariances within the classes. Thus, FDA searches the direction $\vec{w}$ which maximizes the target function S in the following equation:

$$S(\vec{w}) = \frac{\vec{w}^T \cdot M_b \cdot \vec{w}}{\vec{w}^T \cdot M_w \cdot \vec{w}}$$

S can be reduced to a generalized eigenvalue problem and thus, the optimal $\vec{w}$ is the eigenvector corresponding to the largest eigenvalue of $M_w^{-1} \cdot M_b$. Using $\vec{w}$ as the normal vector of a hyperplane yields a linear classifier. However, when considering the complete base of eigenvectors and weighting them with their inverse eigenvalues yields an affine transformation of the feature space. In the transformed feature space the distances can be expected to express similarity w.r.t. the class labels in a much better way. Let us note that though there are more recent and accurate approaches to supervised metric learning, Fisher Faces are still rather popular due to their simplicity.

Relevant Component Analysis (RCA) was introduced in [1] and overcomes several problems of Fisher faces. A first problem of Fisher faces is that it cannot be guaranteed that M_w is invertible. Therefore, RCA introduces an PCA step to eliminate linear dependencies. Another difference is that RCA does not directly optimize the feature space to separate classes. Instead

of objects labeled with classes, RCA requires sets of similar objects called chunks. Therefore, the authors introduce RCA as an unsupervised method. In this paper, we consider RCA as a supervised method because it still requires examples in the form of chunks. For each chunk, the covariance matrix is built and summed up into a matrix corresponding to M_w. Determining the counterpart to M_b in FDA yields another problem. Though we have the information that vectors within the same chunk are similar, we cannot conclude that vectors from different chunks are dissimilar. Therefore, to model the covariance w.r.t. dissimilar objects, RCA considers the covariance matrix M_{all} w.r.t. any object of the data set. The idea behind this solution is the following. For any object o in the data set DB, the majority of objects is usually quite dissimilar. Thus, M_{all} resembles the covariances of dissimilar objects to a much larger extend than those being observed for similar objects.

Though the optimization step of RCA is quite similar to the optimization step in Co-RCA, there is some important differences. While RCA assumes given information about sets of similar objects (chunks), our algorithm does not require any such information. Instead the information that is required for determining M_w is completely derived from other views by co-learning. Another difference is that Co-RCA works on a sample of object pairs being represented as difference vectors. Thus, the covariance information is not related to a mean value but over any distance computation in the complete feature space. Finally, while RCA suffices to compute M_{all} to represent the covariance for dissimilar objects, Co-RCA maintains a dedicated set of example comparisons for dissimilar object pairs.

4 Co-RCA

A multi-view object o is given by an n-tuple $(x_1, \ldots, x_n) \in F_i \times \ldots \times F_n$ of feature vectors drawn from various feature spaces F_i. Each feature space F_i is a subset of $\mathbb{R}^{d_i}$ having the dimensionality of $d_i \in \mathbb{N}$. In the following, we will also refer to F_i as the ith view. To measure the similarity between two vectors in view F_i, we assume the Euclidian distance: $dist(x_i, y_i) = \|x_i - y_i\|$

Our goal is now to find an affine transformation B_i for improving F_i in a way that $dist(x_i \cdot B_i, y_i \cdot B_i)$ yields a better approximation for the similarity of o_x, o_y than $dist(x_i, y_i)$. In particular, we want to decrease the distances of similar objects and increase the distances between dissimilar objects. To achieve this goal, we rely on the same optimization as FDA and RCA:

$$S(\vec{w}) = \frac{\vec{w}^T \cdot M_{dis} \cdot \vec{w}}{\vec{w}^T \cdot M_{sim} \cdot \vec{w}}$$

A major difference of Co-RCA to ordinary RCA are the covariance matrices M_{dis} and M_{sim} which are determined in a different way from the previously discussed methods.

To estimate both matrices, we assume two example sets. The first set T_{sim} consists of multi-view object pairs (o, u) being similar and the second set T_{dis} consists of examples for pairs of dissimilar objects. Note that the sets consists of complete multi-view objects and not of feature vectors because similarity is considered to hold for the complete object without any restriction to a particular view. Given an example set of pairs of multi-view objects, we can now determine the covariance matrices M^i_{sim} and M^i_{dis} in the ith view F_i. Since a covariance matrix is built from feature vectors and not from object pairs, we first of all built the difference vector for each object pair o, u with $o = (x_1, .., x_n)$,$u = (y_1, .., y_n)$ in view F_i: $\delta_i(o, u) = x_i - y_i$. However, since the order of o, u in an object pair is arbitrary and the order of vectors for determining $\delta_i(o, u)$ is not, we also have to consider the inverse vector $\delta_i(u, o) = y_i - x_i$. Thus, we have to derive a set of $2k$ difference vectors from an example set of k object pairs.

Using distance vectors to optimize a feature space has a major impact to the method. A first effect is that resulting data distribution is always point symmetric to the origin. Thus, describing the distribution by a covariance matrix makes sense in most cases. Another important consequence is the amount of distance vectors has the quadratic size of the data set. Thus, it is often required to restrict the optimization to sample sets instead of computing the distribution of the complete set of distance vectors. A final result of using difference vectors is that the optimization is independent from the any locality within the data set. Thus, the information about locations of the compared objects within the feature space is completely discarded and only the information about the relative positioning of both objects is preserved. Since the goal of our method is to find a global affine transformation for the complete feature space, our model cannot cope with local differences in the meaning of the features anyway. Additionally, unlike FDA and RCA, we do not have to make any assumption of the locality of similar objects. Thus, we do not need a compact area in the data space where all objects are considered to be similar to each other such as chunks or classes.

Formally, we can derive M^i_{sim} for the example set T_{sim} in the following way:

$$M^i_{sim}(T_{sim}) = \frac{\sum_{(o,u) \in T_{sim}} \delta_i(o, u) \cdot \delta_i(o, u)^T + \delta_i(u, o) \cdot \delta(u, o)^T}{2 * |T_{sim}|}$$

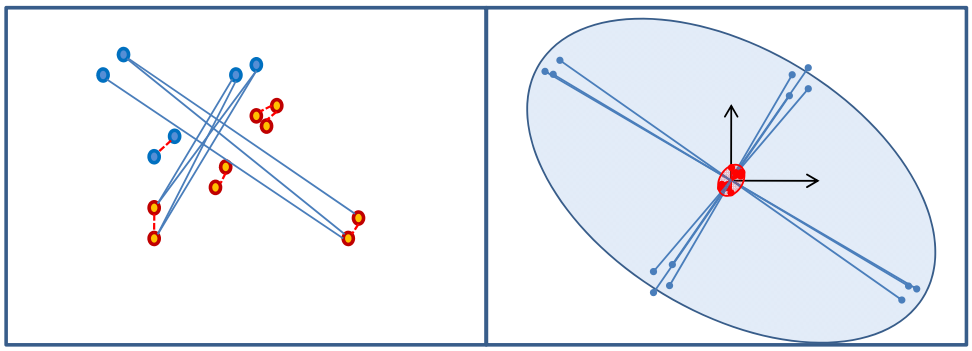


Figure 1: Feature space with depicted closest pairs and farthest pairs(left). Corresponding distribution of distance vectors (right).

M^i_{dis} is determined in the same way by exchanging T_{sim} with T_{dis}.

Now that we know how to determine the matrices based on object pairs, we have to find a way to find meaningful example sets for similar and dissimilar objects. The solution in this paper is based on the following assumption: If two objects have a very small distance in a feature space that is useful for describing similarity, it is very likely that they are semantically similar as well. To find pairs of semantically dissimilar objects, we can use a similar argumentation: If two objects are very different in some feature space, it is very likely that they are really dissimilar to each other w.r.t. the judgement of a human expert. Let us note that the sets of k-closest and k-farthest object pairs do not necessarily contain information about all data objects in the data set. It is very likely that the sample sets contain multiple object pairs containing feature vectors being located in more dense areas. A further important aspect of our method is that the closest object pairs in one feature space are used to optimize the next dimension. Thus, the object pairs being collected in view F_i are used to derive difference vectors in view F_{i+i} where the corresponding distances might be considerably larger. Let us note that the major reason this assumption holds in most cases is that domain experts tend to select meaningful representations which are already selected because they are connected to the task at hand. An expert would not select color features for comparing medical images such as x-Ray or CT images, but grey values, gradients and textures. Thus, using views that do not measure aspects of importance to object similarity will lead Co-RCA to model an unwanted notion of similarity. An illustration for our method to determine covariance matrixes is depicted in figure 1. In the left picture, there is a set with some object pairs being marked as similar(dashed lines) and some object pairs being dissimilar (solid lines). In the right picture, we see the corresponding difference vectors and surrounding ellipses for each distributions to show the general correlations. Though we only have some information given by object pairs, the distribution for similar and dissimilar objects offers a good model to separate the red from the blue vectors on the left side.

To summarize, the example set T_{sim} is extended in the ith view of the data set DB by calculating the k-closest pairs in view i. Analogously, we can record the k pairs of feature vectors having the largest pairwise distance in view i to gather examples for T_{dis}.

The Co-RCA algorithm requires a set of multi-view data objects DB and a sample size k as input. The number of views n needs to be at least two.The output of our algorithm is a set a affine transformation B_i for each view F_i that minimizes the distances of the object pairs in T_{sim} compared to the dissimilar object pairs in T_{dis}. Since the transformation for each view tries to mirror the same set of similarity relations, Co-RCA tries to achieve an agreement of views.

The algorithm starts with determining T_{sim} and T_{dis} from the k-closest and k-farthest pairs in the first view F_0. Then the algorithm enters its main loop where a transformation for the next view F_1 is computed and new elements are added to T_{sim} and T_{dis}. The algorithm terminates when no additional examples can be found among the k closet pairs in any view. We will now explain the steps that are performed in each iteration in greater detail. We start with switching to the next view. When reaching the last view, we return to the first one. To derive the sample sets of difference vectors in the current view, we have to iterate over T_{sim}. For each object pair $(o, u) \in T_{sim}$, we have to compute the difference vector $\delta_i(o, u)$ and use it compute M^i_{sim}. Correspondingly, we iterate over T_{dis} and calculate M^i_{dis}. Let us note that it is important

```
Co-RCA ( MR_Database DB, k)
    rep := 1
    nochange := 0
    T_sim := k-closet-pairs(DB[rep],k)
    T_dis  := k-farthest-pairs(DB[rep],k)
    B[]    //result transformation for all reps
    while( |T_sim|+| T_dis| > old or nochange ≠ n)
        rep := (rep +1)mod n
        M_sim := Covariance(T_Sim, rep)
        M_dis  := Covariance(T_dis rep)
        B[rep] :=RCA (M_sim , M_dis,DB[rep])
        newDB := RCA[rep](DB[rep])
        T_sim := T_sim ∪ k-closet-pairs(newDB,k)
        T_dis  := T_dis ∪ k-farthest-pairs(newDB,k)
         if |T_sim|+| T_dis| = old
             nochange := nochange+1
        else
             nochange := 0
         endif
        old:=|T_sim|+| T_dis|
    endwhile
    return B[]
```

Figure 2: Pseudocode of Co-RCA for n views.

that T_{sim} and T_{dis} contain examples being close in other views but not in F_i. Thus, F_i is optimized to contain similarity relationships from the other views. Then, we employ both matrices in the optimization step being similar to RCA. The result is an affine transformation B_i. The step starts with performing an PCA on M^i_{sim} to make sure it is invertible. After applying the PCA, we solve the eigenvalue problem on $(M^i_{sim})^{-1} \cdot M_{dis}$. The resulting affine transformation B_i is composed of the weighted eigenvectors of $(M^i_{sim})^{-1} \cdot M_{dis}$ where each eigenvector v_j is weighted with the inverse square root of the corresponding eigenvalue λ_j: $\frac{1}{\sqrt{\sqrt{\lambda_j}}}$. This transformation is also known as whitening transformation. The transformation B_i is now applied to the the view F_i. Then, the k-closest and the k-farthest pairs of object are determined in the transformed feature space leading to new example sets $T_{\grave{sim}}$ and $T_{\grave{sim}}$ which are joined to the already known example sets T_{sim} and T_{dis}. Let us not that it is possible that $T_{\grave{sim}} \subset T_{sim}$ or $T_{\grave{dis}} \subset T_{dis}$, indicating that the view is already consistent with the example sets. The algorithm stops when all views are consistent with the example sets, i.e. no new examples can be found to extend the example pairs. Therefore, it is necessary that there are n consecutive iterations not adding any new examples. The pseudo code of Co-RCA is depicted in figure 2.

So far the result of Co-RCA is an affine transformation B_i for each feature space F_i. However, in order to employ the results in data mining, it is necessary to integrate the transformations into the data mining algorithm. The simplest way to do this is to add the feature transformations B_i into the preprocessing step of the knowledge discovery process and only work with the transformed feature spaces. Another option is to use the original features spaces and work with Mahanalobis distance using $B_i^T \cdot B_i$. A further problem is how to employ all representations for determining similarity. The simplest solution to do this would be to employ only a single representation. Since the similarity statements should agree, similarity should be reflected in all representations in a similar way. However, though Co-RCA can improve the similarity relationships in the particular representation, a less useful relation might not be transformed into a very useful one by means of a affine basis transformation. Thus, it is very likely that some representations yield more information than others and using a single representation yields the risk of selecting a less informative one. We therefore argue to employ all representations by means of a simple multi-represented distance measure such as the normalized sum of distances:

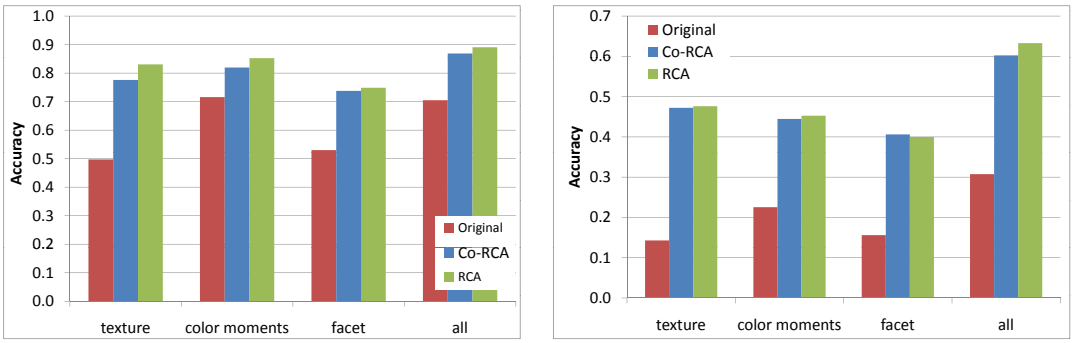
$$dist_{norm}(o,u) = \sum_{1=i}^{n} w_i \cdot \|x_i - y_i\|$$

where w_i is a weight factor reflecting the importance of representation i. If no weight is available all representations can be weighted equally, setting $w_i = 1$. Having a multi-view distance measure allows us to use multiple distance-based data mining algorithms like density-based clustering such as DBSCAN [9]. Furthermore, the resulting affine transformations can be easily integrated into multi-view kernel learning.

5 Experimental Evaluation

In this section, we present the results of preliminary experiments with Co-RCA. We test our method on two image data sets where each image is categorized into exactly one class. The first data set (Conf183) contains 183 pictures belonging to 35 classes which were photographed during two sightseeing trips. The second data set S4B consists of 1743 photographs from 80 classes. We extracted color moments, texture features [13] and facet-orientations [5] for each image in both data sets. Thus, both experiments were done on 3 representations.

As comparison partner, we chose the original features spaces and their linear combination. Furthermore, we wanted to compare Co-RCA to the closest supervised method which is RCA. As chunks for training RCA, we employed the class label. The classes in both data sets have all a limited amount of instances and thus, they are well-suited to be used as chunks. We employed RCA to see how closely the result of the unsupervised Co-RCA would get to a supervised method. The sample size k for



(a) Conf 183 (b) S4B

Figure 3: 1NN classification accuracies being achieved on each representation and their linear combination.

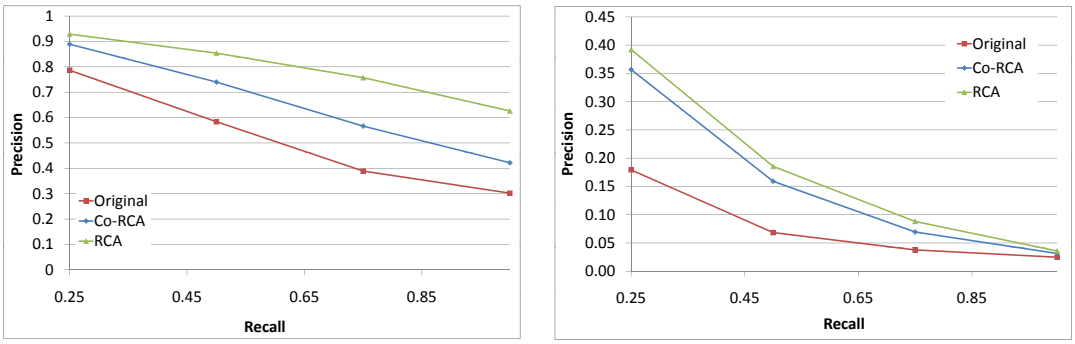
Co-RCA was set to 250 for the smaller data set conf183 and to 5000 for the larger data set S4B. Let us note that though these sample sizes seem to be rather large, they are still comparably small to the number of all possible difference vectors. All methods were implemented in Java 1.7 and tests were run on a dual core (3.0 Ghz) workstation with 2 GB main memory.

A first test for the quality of a feature space is nearest neighbor classification. In other words, we test in how many cases the closest neighbor in a given data representations is from the same class as the query object. To avoid that the query object is part of the database itself, we tested using stratified 10-fold cross-validation. Let us note that we trained RCA and Co-RCA on the complete data set instead of the limited training set within each fold. This, approach is viable for Co-RCA because it is a completely an unsupervised method. For RCA, the results might slightly overfit. However, since RCA is only a comparison partner to the unsupervised Co-RCA, the results are sufficient to show how close Co-RCA might get to RCA. The results are shown in figure 3. For the small and relatively easy conf183 data set. The 1NN classifier on the original feature set without any optimization achieved classification accuracies between 50 % and 70 %. The combination off all three representations did not yield a performance advantage but performed similar to the best single representation, i.e. color moments. As expected the supervised RCA achieved the best results for all representations and the combination off all representations improved the accuracy by 4 %. The unsupervised Co-RCA achieved between 2% and 5 % less classification accuracy than the supervised RCA for all representations. For the best result achieved by combining all representations, the performance was only 2% less accurate than for the supervised method. The plot for the larger and more difficult S4B data set shows similar results. The 1NN classification on the original feature spaces did not yield convincing results. However, the unsupervised Co-RCA managed to get very close to the supervised RCA. For the combination of all three representations, Co-RCA achieved only 3 % less accuracy. To conclude, by exploiting the structure of 3 different views of the data set, it was possible to considerably improve the similarity of the nearest neighbors in the underlying feature space.

In the next experiment, we measure the performance w.r.t. all similar objects in the data set and not just for the closest ones. Thus, we pose a ranking query for each data object and count the number of result objects belonging to the same class. The resulting precision recall graph is averaged over all possible query objects in the database and displays the number of true hits (precision) in the result set containing at least x % of the similar objects in the database (recall). For both graphs, we can observe a similar behavior. Though Co-RCA cannot achieve the high precision levels of RCA, it is still able to considerably improve the precision for any measured level of recall. For the more difficult data set S4B, the graph is even considerably closer to the supervised benchmark than to the default case of the linear combination of the original feature spaces.

6 Conclusions

In this paper, we presented ongoing work in unsupervised feature space optimization for multi-view data. The core idea of our new method Co-RCA is to combine co-training and distance learning methods to generate a multi-view data space yielding a better model for object similarity. The leverage of our unsupervised method is the observation that very close and very distant objects in a meaningful feature space usually indicate pairs of objects which are semantically similar or dissimilar as well. Based on this type of information, it is possible



(a) Conf 183 (b) S4B

Figure 4: Precision-recall graph for the joint representations for the three types of feature spaces.

to iteratively select examples for similar and dissimilar object pairs from all representations. In a co-training algorithm we iteratively optimize the feature spaces and select new examples. Our experiments indicate that our new algorithm Co-RCA can generate feature transformations of almost comparable quality as the supervised approach our methods is built on, RCA.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of the 20th International Conference on Machine Learning (ICML), Washington, DC, USA*, pages 11–18, 2003.

[2] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with Co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT), Madison, WI*, pages 92–100, 1998.

[5] G. Chinga, O. Gregersen, and B. Dougherty. Paper surface characterisation by laser profilometry and image analysis. *Journal of Microscopy and Analysis*, 84:5–7, 2003.

[6] P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314, 1994.

[7] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman & Hall/CRC, 2nd edition, 2001.

[8] J. Davis, B. Kulis, S. Sra, and I. Dhillon. Information-theoretic metric learning. In *in NIPS 2006 Workshop on Learning to Compare Examples*, 2007.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR*, 1996.

[10] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[11] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighborhood component analysis. In *Advances in Neural Information Processing Systems*, pages 513–520. MIT Press, 2004.

[12] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[13] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Speech and Audio Processing*, 3(6):6103–623, 1973.

[14] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[15] J. B. Tenenbaum, V. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[16] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*, Cambridge, MA, USA, 2006. MIT Press.

[17] B. Yan and C. Domeniconi. Subspace metric ensembles for semi-supervised clustering of high dimensional data. In *Proceedings of the 10th European Conference on Principles of Knowledge Discovery and Data Mining (PKDD), Berlin, Germany*, 2006.

[18] L. Yang. An overview of distance metric learning. Technical report, Department of Computer Science and Engineering, Michigan State University, 2007.

[19] D. Zhai, H. Chang, S. Shan, X. Chen, and W. Gao. Multi-view metric learning with global consistency and local smoothness. In *ACM Trans. on Intelligent Systems and Technology*, 2011.

New Subspace Clustering Problems in the Smartphone Era

Dimitrios Gunopulos
University of Athens
dg@di.uoa.gr

Vana Kalogeraki
Athens Univ. of Economics and Business
vana@aueb.gr

Abstract

We describe current and future work on the problem of subspace clustering and its application in the current environment where powerful, programmable mobile communication and computing devices proliferate. We show that a smartphone cloud computing infrastructure is feasible. We focus on specific variations of the subspace clustering problem, namely the need to develop algorithms that take into account input from the user and the need for privacy preserving algorithms.

1 Introduction

The proliferation of powerful, programmable mobile devices along with the availability of wide-area connectivity has provided a powerful platform to sense and share location, motion, acoustic and visual data. People carry these devices with them everywhere. In the early 2010 period, it is reported that 42.7 million people in the U.S. alone own smartphones, 18 percent higher than 4 months prior [1]. These devices are outfitted with a wide array of sensing capabilities such as GPS, WiFi, microphones, cameras and accelerometers.

These mobile devices allow always on, full featured connectivity, and they bring a true revolution in the way people connect and use the web. We are now entering an era where people can share aspects of their lives online, creating virtual communities, and becoming both data producers and data consumers at the same time. With smartphones becoming the mobile platform of choice, a number of applications have emerged in a wide variety of domains such as personal life, travel and work. Examples include traffic monitoring for real-time delay estimation and congestion detection [2], location-based services such as personalized weather information, locating areas of good WiFi connectivity, location-based games or receiving spatial alarms upon the arrival to a reference time point [3], and social networking applications for sharing photos and personal data with family and friends [4].

The data collection capabilities that are embedded in these mobile devices coupled with the fact that their processing power, networking capability and resource capacity is increasingly rapidly, has make these devices a very attractive and significant platform for *smartphone cloud computing*[5].

2 Desiderata for new mobile systems

Using smartphones to create a cloud computing infrastructure opens up new opportunities for mobile computing. The exploitation of these capablities provides excellent opportunities to sense data at locations where people travel and provides many possible applications to study users' behaviors [6] while the users themselves provide automatic mobility for sensors.

Although today's mobile devices offer a powerful sensing and a versatile computing platform, developing applications to take advantage of them is challenging due to a number of issues that must be addressed. We have established the following design considerations for the development of the mobile systems and applications of the future:

1. **Distributed Computation:** The system should be distributed, so that it scales with the number of users and does not have a single point of failure.
2. **Ease of Development:** The MapReduce framework has been successfully used to develop large scale distributed applications, and its use can significantly simplify and ease the programming and deployment costs of distributed mobile systems in the future.
3. **Security and Privacy:** The system should leverage and promote user collaboration; however it should safeguard the individual user resources and protect the privacy of the individual user data.
4. **Ease of Use:** The application must be easy to use, so that a user can receive and understand the results easily. This means that the solutions must be presented in an intuitive and easy to comprehend way, but also that the user should have the ability to tune the solutions based in his on her needs, in a way as simple as possible.
5. **Connectivity and failures:** The problem of handling spotty connectivity and failures gracefully, which is a problem afflicting all mobile systems.

We note here, that some traditional considerations of mobile distributed systems become less important in our setting. For example, devices such as smart-phones and PDAs naturally use a lot of power and therefore are typically charged regularly–daily. Power consumption, a very important problem in traditional sensor systems, while still important, is less important for such mobile devices.

We should also emphasize here that one difference between a cloud of smartphones and the traditional IT infrastructure is how privacy issues can be addressed; data does not have to be communicated and stored into remote servers in a data center, data is rather kept privately on the user phones and communicated only when certain conditions arise or is needed by the user. Finally the cost model is different as well. The mobile cloud is an alternative to traditional data centers where servers are usually dedicated to particular applications, as mobile clouds have low infrastructure costs, low management overhead and exhibit greater flexibility.

3 The MapReduce Framework

To fully take advantage of the available resources, a simple, but versatile system which eases development deployment of software is essential. To this end we have developed Misco [7], a system that implements a MapReduce framework targeted for smartphones and mobile devices. The MapReduce framework [8] is a flexible, distributed data processing framework designed to automatically parallelize the processing of long running applications on petabyte sized data in clustered environments. The MapReduce programming model [8] supports the weak connectivity model of computations across open networks, such as mobile networks, which makes it very appropriate for a mobile setting.

MapReduce provides two functional language primitives: *map* and *reduce*. The map function is applied on a set of input data and produces intermediary $< key, value >$ pairs, these pairs are then grouped into R partitions by applying some partitioning function (e.g. $hash(key)$ MOD R). All the pairs in the same partition are passed into a reduce function which produces the final results.

The MapReduce programming model provides a simple way to split a large computation into a number of smaller tasks; these tasks are independent of each other and can be assigned on different worker nodes to process different pieces of the input data in parallel.

Application development is greatly simplified as the user is only responsible for implementing the map and reduce functions and the MISCO system handles the scheduling, data flow, failures and parallel execution of the applications.

The Misco system [7] is a MapReduce implementation that runs on mobile phones. Misco comprises a *Master Server* and a number of *Worker Nodes*. In [9] we presented the system we have developed that allows for easy development of applications on networks of mobile smartphones that employ data clustering as a fundamental data analysis operation, while at the same time safeguarding individual user resources and preserving data privacy. Although we focus on clustering, our approach is general; it shows that the development of efficient techniques for sophisticated data analysis enables interesting and novel applications. We demonstrate the usefulness of this system by implementing an application that identifies areas of high WiFi connectivity.

4 Novel Applications

Research in developing sophisticated data analysis techniques in the smartphone cloud computing paradigm is fundamental for building novel and useful applications that fully exploit this setting. We expect that the development of efficient algorithms for general purpose data analysis and data mining tasks will enable new applications that not only will leverage the proliferation of smartphones but also further accelerate their use.

The main thrust of our work is to allow users to collaborate through their mobile phones and combine their individual data to find such areas while maintaining their privacy.

In this work we initiate work to develop subspace clustering algorithms that run on smartphone clouds. We describe the problems where such techniques are important to develop, and investigate the technical challenges that have to be overcome in order to develop efficient techniques.

4.1 The setting: We assume the following setting: Users use their smartphones to connect to the system, and also to store their local data. Each user can decide what types of data should be stored, but generally such data have spatiotemporal charcteristics. Such data include location data which record the movement of the user, and possibly geo- and time-tagged information such as notes, files, pictures. The location data are sequences of GPS traces over time [10]. We can also expect that these traces can be joined to metadata describing specific locations (e.g. stores, landmarks, etc).

We note here that the problem of enriching the GPS traces with additional information, which can also be described as an event discovery process, is of independent interest, but orthogonal to the problem we describe here; thus for this work we simply assume that

we can take advantage of it if available.

The users use their smartphone to access the system, ask queries, receive recommendations, analyze the data, and make their data available, if this is required or allowed by the user.

4.2 Applications: To motivate our approach, we briefly describe two different applications or the system.
Identifying locations that are frequently visited together: consider for example a transit authority which plans its bus routes and wants to know whether there is demand for possible new bus routes. In such a scenario, one is interested in finding out if there is a large number of people that are having the same trajectory (possibly passing by the same set of landmarks) without disclosing their individual trajectories.
Recommending popular locations or tourist spots for travelers or locals: consider a guide application that suggests possible tours through a city, based on a user's preferences and by considering the spots visited by other people with similar interests or history.

4.3 Problems: In both applications the technical problem is how to find groups of locations that have been consistently popular for a large group of people and also, groups of people that have all been in the same subset of locations in approximately the same order. This motivation essentially dictates the use of subspace clustering algorithms to aggress the problem. Formally, we consider the following problems:
Problem 1: Frequent Locations Problem *Given a set of people, and a trace of locations for each person, find sets of locations that have been visited by a large fraction of the people.*

We note that the result of Problem 1 can be a very large set of possible sets of locations. This can create problems both because the large number of sets makes the goal of creating an efficient algorithm unrealistic, but also because the large size of the results makes very difficult for the user to understand them. Therefore, we maintain that it is imperative that user input be used to guide the algorithms and focus the results.
Problem 2: User Constrained Frequent Locations Discovery *Given a set of people, and a trace of locations for each person, find sets of locations that have been visited by a large fraction of the people, while satisfying user constraints that are given on the locations (one location must appear, or two locations have to appear either together or separately).*

We note that additional information that may be available about the users can also be exploited in this setting. In fact, the use of different sets of attributes in a clustering can potentially identify interesting subgroups of users.

Finally, we consider a variation of Problem 1, which specifies that not only the set of the locations visited must be the same, but the order must be the same (perhaps approximately) as well.
Problem 3: Frequent Subsequence Discovery *Given a set of people, and a trace of locations for each person, find sets of locations that have been visited by a large fraction of the people in the same order.*

5 Related work

5.1 Subspace clustering: There is a large body of work on subspace clustering, and several algorithms have been proposed in the literature. [11] gives a good review and a taxonomy of the algorithms. We note that subspace algorithms come in two varieties: algorithms that allow overlapping clusters (and therefore an object can be in more that one clusters) and algorithms that do not allow overlapping clusters. For the problems we consider, algorithms that allow overlaps are much more useful, because it can clearly be the case that due to its movement an object (or a person) can go through the locations in more that one frequent sets.

5.2 Clustering in The MapReduce paradigm: [12] show that co-clustering algorithms can be efficiently implemented in the MapReduce framework. Their implementation does not consider mobile systems and their constraints however. Our recent work shows that general clustering algorithms (for example, K-Means), can be implemented in the Misco framework.

5.3 Distributed and privacy preserving Association Rule mining: The problem of association rule mining is very closely related to both the problems we consider and to some density based formulations of subspace clustering. There has been significant amount of work on how to find frequent itemsets when the data are distributed to different sites and we want to preserve the privacy of the users, i.e. the other users should not get any additional information about the data of a given user. Generally algorithms fall into two categories, cryptography-based algorithms that require a lot of computation ([13], [14]) or randomization based approaches ([15]). The randomization approach has also been used to anonymize trajectory data as well [16].

6 Building novel applications in the mobile environment

The problems we identify above motivate future research along the following lines:
Develop subspace clustering algorithms for mobile systems using the MapReduce framework.

In this context, the Misco system gives an efficient testbed to develop and test such algorithms. Current work shows that iterative, non-overlapping cluster methods can be efficiently implemented in a distributed system. Several questions have to be addressed however. One of the most important is the development of lightweight privacy preserving mechanisms. We note here that cryptography based approaches may prove difficult to adapt to the mobile environment due to their computational requirements.

Develop subspace clustering algorthms that incorporate user constraints. More specifically, we will investigate different mechanisms to incorporate user constraints. These constraints can be used to differentiate between alternative clusterings, if used after the clustering process, or can be used to guide the clustering approach, if used during the clustering.

Distributed techniques for sequential pattern discovery. We note here that finding frequent constrained sequential patterns is a problem yet it has important connections to the subspace clustering algorithms.

Next we outline future research in addressing these topics, and propose extensions to existing work. Current work focuses on non-overlapping clustering approaches and considers extensions of the iterative clustering framework to take into account constraints and identify interesting subspaces. The non-overlaping clustering approach (see for example PROCLUS [17] and related work), can be incorporated in our framework following the same approach that was taken for computing the K-Means algorithm.

The more general, overlapping-cluster approach typically uses a density based formulation. The fundamental problem here is that we have to develop a mechanism that can use user input to guide the search among subspaces since the number of possible subspaces is large. In addition, we have to understand the impact of problems such as sampling (of the dataset and of the solution space), the effect of horizontal vs vertical partitioning of the way data is stored, and of minimizing the information shared by the users. Finally, related work [18] has shown that efficient indexing structures can be used with good benefit to identify frequent subspaces, however the problem of developing algorithms for the mobile environment is novel.

References

[1] "comscore reports," http://www.comscore.com/.

[2] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *SenSys 2009*.

[3] B. Bamba, L. Liu, A. Iyengar, and P. S. Yu, "Distributed processing of spatial alarms: A safe region-based approach," in *ICDCS*, Washington, DC, 2009, pp. 207–214.

[4] E. Miluzzo, C. Cornelius, A. Ramaswamy, T. Choudhury, Z. liu, and A. Campbell, "Darwin phones: the evolution of sensing and inference on mobile phones," in *Mobisys 2010, June 15-18, 2010, San Fransisco, CA*.

[5] ReadWriteWeb, "Exploring how to build a cloud of smartphones," http://www.readwriteweb.com/cloud/2010/08/nokia-explores-how-to-build-a.php.

[6] E. Miluzzo, N. D. Lane, K. Fodor, R. A. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *SenSys*, 2008.

[7] A. J. Dou, V. Kalogeraki, D. Gunopulos, T. Mielikäinen, and V. H. Tuulos, "Scheduling for real-time mobile mapreduce systems," in *DEBS*, 2011.

[8] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *OSDI, San Francisco, CA, USA*, Dec 2004, pp. 137–150.

[9] A. Dou, V. Kalogeraki, D. Gunopulos, T. Mielikinen, V. Tuulos, S. Foley, and C. Yu, "Data clustering on a network of mobile smartphones," in *SAINT 2011*.

[10] D. Zeinalipour-Yazti, C. Laoudias, M. I. Andreou, and D. Gunopulos, "Disclosure-free gps trace search in smartphone networks," in *Mobile Data Management (1)*, 2011.

[11] E. Müller, S. Günnemann, I. Färber, and T. Seidl, "Discovering multiple clustering solutions: Grouping objects in different views of the data," in *ICDM*, 2010.

[12] S. Papadimitriou and J. Sun, "Disco: Distributed co-clustering with map-reduce: A case study towards petabyte-scale end-to-end mining," in *ICDM*, 2008.

[13] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," in *DMKD*, 2002.

[14] M. G. Kaosar and X. Yi, "Semi-trusted mixer based privacy preserving distributed data mining for resource constrained devices," *CoRR*, vol. abs/1005.0940, 2010.

[15] A. V. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules," *Inf. Syst.*, vol. 29, no. 4, pp. 343–364, 2004.

[16] M. E. Nergiz, M. Atzori, Y. Saygin, and B. Güç, "Towards trajectory anonymization: a generalization-based approach," *Transactions on Data Privacy*, vol. 2, no. 1, pp. 47–75, 2009.

[17] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park, "Fast algorithms for projected clustering," in *SIGMOD Conference*, 1999, pp. 61–72.

[18] M. R. Vieira, P. Bakalov, and V. J. Tsotras, "Flextrack: A system for querying flexible patterns in trajectory databases," in *SSTD*, 2011, pp. 475–480.

Multiple Clustering Views via Constrained Projections *

Xuan Hong Dang, Ira Assent† James Bailey‡

Abstract

Clustering, the grouping of data based on mutual similarity, is often used as one of principal tools to analyze and understand data. Unfortunately, most conventional techniques aim at finding only a single clustering over the data. For many practical applications, especially those being described in high dimensional data, it is common to see that the data can be grouped into different yet meaningful ways. This gives rise to the recently emerging research area of discovering alternative clusterings. In this preliminary work, we propose a novel framework to generate multiple clustering views. The framework relies on a constrained data projection approach by which we ensure that a novel alternative clustering being found is not only qualitatively strong but also distinctively different from a reference clustering solution. We demonstrate the potential of the proposed framework using both synthetic and real world datasets and discuss some future research directions with the approach.

1 Introduction.

Cluster analysis has been widely considered as one of the most principal and effective tools in understanding the data. Given a set of data observations, its objective is to categorize those observations that are similar (under some notion of similarity) into the same cluster whilst separating dissimilar ones into different clusters. Toward this goal, many algorithms have been developed by which some clustering objective function is proposed along with an optimization mechanism such as k-means, mixture models, hierarchical agglomerative clustering, graph partitioning, and density-based clustering. A general observation with these algorithms is that they only attempt to produce a single partition over the data. For many practical applications, especially those being characterized in high dimensional data, this objective seems to be not sufficient since it is common that the data observations can be grouped along different yet equally meaningful ways. For example, when analyzing a document dataset, one may find that it is possible to categorize the documents according to either topics or writing styles; or when clustering a gene dataset, it is found that grouping genes based on their functions or structures is equally important and meaningful [4]. In both these applications and many other ones, we may see that the natural structure behind the observed data is often not unique and there exist many alternative ways to interpret the data. Consequently, there is a strong demand to devise novel techniques that are capable of generating multiple clustering views regarding the data.

In the literature, several algorithms have been developed to seek alternative clusterings and it is possible to categorize them into two general approaches: seeking alternative clusterings simultaneously [13, 8, 22] and seeking alternative clusterings in sequence [3, 6, 10, 9]. In the former approach, all alternative clusterings are sought at the same time whereas in the latter one, each novel clustering is found by conditioning on all previous clusterings. From a modeling view point, the latter approach has a major advantage that it limits the number of cluster parameters needed to be optimized concurrently.

In this paper, we develop a novel framework to find multiple clustering views from a provided dataset. Given the data and a reference clustering $C^{(1)}$ as inputs, our objective is to seek a novel alternative clustering that is not only qualitatively strong but also distinctively different from $C^{(1)}$. The proposed algorithm achieves this dual objective by adopting a graph-based mapping approach that preserves the local neighborhood proximity property of the data and further conforms the constraint of clustering independence with respect to $C^{(1)}$. Though our research can be categorized into the second approach of sequentially seeking alternative clusterings, it goes beyond the work in the literature by further ensuring that the geometrical proximity of the data is retained in the lower mapping subspace and thus naturally reveals clustering structures which are often masked in the high dimensional data space. We formulate our clustering objective in the framework of a constrained eigendecomposition problem and thus it has a clear advantage that a closed form solution for the learning subspace always exists and is guaranteed to be globally optimal. This property contrasts the proposed

*Part of this work has been supported by the Danish Council for Independent Research - Technology and Production Sciences (FTP), grant 10-081972.

†Dept. of Computer Science, Aarhus University, Denmark. Email: {dang,ira}@cs.au.dk.

‡Dept. of Computing and Information Systems, The University of Melbourne, Australia. Email: baileyj@unimelb.edu.au.

framework to most existing algorithms, which solely focus on optimizing the single condition of decorrelation and may only achieve a local optimal subspace. We demonstrate the potential of the proposed framework using both synthetic and real world data and empirically compare it against several major algorithms in the literature. Finally, some future studies for our framework are discussed and we shortly suggest some potential approaches to deal with them.

2 Related Work.

Though being considered related to subspace clusterings [1, 15, 16, 28, 20], the problem of discovering multiple alternative clusterings is relatively young and recently it has has been drawing much attention from both data mining and machine learning communities. In [19], the authors give an excellent tutorial regarding different approaches to the problem. In this section, we adopt the taxonomy that can generally divide the majority of work in the literature into two approaches: those seeking alternative clusterings concurrently and those seeking alternative clusterings in sequence, and briefly review the algorithms that closely related to our research in this paper.

In the first approach, two algorithms named Dec-kmeans and ConvEM are developed in [13] to find two disparate clusterings at the same time. In Dec-kmeans, the concept of representative vectors is introduced for each clustering solution. Subsequently, the objective function of the k-means method is modified by adding terms to account for the orthogonality between mean vectors of one clustering, with respect to the representative vectors of the other. In the ConvEM algorithm, a similar approach is applied by assuming that the data can be modeled as a sum of mixtures and this work associates each mixture with a clustering solution. This leads to the problem of learning a convolution of mixture distributions by which the expectation maximization method can be employed to find the distributions' parameters. Another algorithm called CAMI based on mixture models is developed in [8]. However, instead of trying to orthogonalize two sets of cluster means, CAMI takes into account a more general concept of mutual information to quantify for the decorrelation between two clustering models. The algorithm thus attempts to minimize this quantity while at the same time maximizing the likelihood of each respective model.

In the second approach, an algorithm named COALA is proposed in [3]. Given a known clustering, COALA generates a set of pairwise cannot-link constraints and it attempts to find a disparate data partition by using these constraints within an agglomerative clustering process. The NACI algorithm developed in [9] takes a different approach purely stemming from information theory. Its clustering objective is thus to maximize the mutual information between data instances and cluster labels of the alternative clustering while minimizing such information between that alternate and the provided clustering. However, instead of using the traditional Shannon entropy [5], this work is developed based on the use of Renyi's entropy, with the corresponding quadratic mutual information [14, 24]. Such an approach allows the MI to be practically approximated when combined with the non-parametric Parzen window technique [23]. Recently, this dual-optimized clustering objective is also exploited in work [21] with an iterative approach, in contrast to the hierarchical technique adopted in [9]. Another line in the second approach are the algorithms developed in [6, 10, 25] which address the alternative clustering problem via the use of subspace projection. Work in [6] develops two techniques to find an alternative clustering using orthogonal projections. Intuitively, one can characterize a data partition by a set of representatives (e.g., cluster means). It is then possible to expect that a dissimilar partition might be found by clustering the data in a space orthogonal to the space spanned by such representatives. In the first algorithm in [6], clustering means learnt from a given partition are used as representatives, whilst in the second algorithm, principal components extracted from such centroid vectors are used. A similar approach is developed in [10] in which the transformation is applied on the distance matrix learnt from the provided clustering. Compared to the two methods developed in [6], this work has a benefit that it can further handle the problem of which the data dimension can be smaller than the number of clusters (e.g., spatial datasets). Another approach based on data transformation is developed in [25]. This work attempts to transform the data such that data instances belonging to the same cluster in the reference clustering are mapped far apart in the newly transformed space. Our research in this paper is also to adopt the data projection approach. However, we go beyond these related algorithms by ensuring that the novel projection subspace is not only decorrelated from the provided clustering solution, the local similar property of the data is also preserved in the projection subspace to strongly support uncovering a novel clustering structure behind the data.

3 Problem Definition.

We define our problem of generating multiple clustering views from the data as follows.

Given a set $\mathcal{X}$ of d-dimensional data instances/observations $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ and a clustering solution $C^{(1)}$ (i.e., a data partition over $\mathcal{X}$ found

by any clustering algorithm) over $\mathcal{X}$, we seek an algorithm that can learn $C^{(2)}$, a novel clustering solution from $\mathcal{X}$, whose clusters $C_i^{(2)}$ satisfy $\bigcup_i C_i^{(2)} = \mathcal{X}$ and $C_i^{(2)} \cap C_j^{(2)} = \emptyset$ for $\forall i \neq j;\ i,j \leq K$, where K is the number of expected clusters in $C_i^{(2)}$. The quality of the alternative clustering should be high and also be distinctively different from $C^{(1)}$. We will here mostly focus on the case where only a single reference clustering is provided but the framework can be straightforwardly extended to the general case of generating multiple alternative clusterings.

4 The Proposed Framework.

4.1 Subspace Learning Objective with Constraint. In generating an alternative clustering, our study in this work makes use of a subspace learning approach. Given $C^{(1)}$ as a reference clustering, we aim to map the data from the original space into a new subspace in which it is uncorrelated from $C^{(1)}$ whereas the mapping also well captures and retains certain properties of the data. The dimension of the subspace is usually smaller than the original one and thus the clustering structure behind the data can be more easily uncovered. Clearly, for our clustering problem, we would like that if two instances $\mathbf{x}_i$ and $\mathbf{x}_j$ are close in the original space, they should be mapped also close to each other in the lower dimensional space. This idea is also behind several methods including Local Linear Embedding [26], Laplacian Eigenmap [18] and Locality Preserving Projection [12]. However, it is worth mentioning that these algorithms are naturally developed to learn a single manifold embedded in a high dimensional space and to find ways to represent it in an optimal subspace. Our research in this work, despite sharing a similar mapping objective, is clearly different as we aim to seek a set of clusters from the data and more importantly, further require the mapping data to be uncorrelated from one or more reference clusterings.

Following this subspace learning approach, we formulate our problem using graph theory. Let $G = \{V, E\}$ be an undirected graph, where $V = \{v_1, \ldots, v_n\}$ is a set of vertices and $E = \{e_{ij}\}$ is a set of edges, each connecting two vertices (v_i, v_j). A vertex v_i corresponds to a data instance $\mathbf{x}_i$ in the dataset $\mathcal{X}$ and the edge e_{ij} between v_i and v_j exists if the respective points $\mathbf{x}_i, \mathbf{x}_j$ are close to each other. Under this setting, the closeness between $\mathbf{x}_i$ and $\mathbf{x}_j$ can be defined using the ℓ-nearest neighbor concept. Specifically, we define $\mathbf{x}_i$ to be close to $\mathbf{x}_j$ if it is among the ℓ-nearest neighbors of $\mathbf{x}_j$ or vice versa. Moreover, for the edge connecting two respective vertices v_i, v_j of such $\mathbf{x}_i$ and $\mathbf{x}_j$, we associate K_{ij} (computed by the Gaussian function [9]) as a measure of their closeness degree. For two vertices that are not connected, the respective K_{ij} is set to zero. Consequently, we denote K as the $n \times n$ matrix having K_{ij} as its elements and it is easy to observe that K is symmetric and typically sparse, since each vertex is only connected to a limited number of its nearest neighbors.

Given the weight matrix K derived from the graph G and a reference clustering $C^{(1)}$, our objective is to learn a novel set $\mathcal{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$, where each $\mathbf{y}_i \in R^q$ is the mapping of $\mathbf{x}_i \in R^d$, *via a projection matrix* F, that optimally retains the local neighborhood proximity of the original data yet taking the decorrelated requirement over the reference solution $C^{(1)}$ into account. Essentially, let us denote X and $Y = F^T X$ two matrices respectively having $\mathbf{x}_i$'s and $\mathbf{y}_i$'s as their column vectors and let $\mathbf{f}$ be a column in F, then we can consider $\mathbf{f}$ as a transformation vector that *linearly* combines X's dimensions into a 1-dimensional vector $\mathbf{y}^T = \{y_1, \ldots, y_n\} = \mathbf{f}^T X$. That means $\mathbf{y}$ is one feature in the mapping space Y. We are now able to define our subspace learning task with the following optimization function:

$$\arg\min_{\mathbf{f}} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\mathbf{f}^T \mathbf{x}_i - \mathbf{f}^T \mathbf{x}_j)^2 K_{ij} \quad \text{s.t. } S^T X^T \mathbf{f} = 0 \tag{4.1}$$

in which S is a feature subspace that best captures the reference solution $C^{(1)}$ and adding the constant $1/2$ does not affect our optimization objective. The constraint $S^T X^T \mathbf{f} = 0$ (or equivalently $S^T \mathbf{y} = 0$) is crucial since it ensures that the mapping dimension is independent from S. Also notice that here we assume X is projected onto $\mathbf{f}$ to form an R^1 optimal subspace, yielding 1-dimensional vector $\mathbf{y}^T = \{y_1, \ldots, y_n\}$. The generalization to q optimal dimensions will be straightforward once we derive the solution for $\mathbf{f}$ and subsequently $\mathbf{y}$.

Observing from Eq.(4.1) that if K_{ij} is large (implying $\mathbf{x}_i$ and $\mathbf{x}_j$ are geometrically close in the original space), the objective function will be large if the two respective points $y_i = \mathbf{f}^T \mathbf{x}_i$ and $y_j = \mathbf{f}^T \mathbf{x}_j$ are mapped far apart. Therefore, finding an optimal vector $\mathbf{f}$ that minimizes Eq.(4.1) is equivalent to optimally retaining the local proximity of the data, subject to the constraint $S^T X^T \mathbf{f} = 0$ to ensure the decorrelation from $C^{(1)}$.

4.2 Fisher Linear Discriminant for S Selection. We now discuss how to select S, a subspace that best captures the provided clustering solution $C^{(1)}$. That means the clusters as components of $C^{(1)}$ can be well discriminated when data is represented in S. In achieving this goal, the Fisher's linear discriminant (FDA) can be a natural choice. Briefly, FDA is a

supervised learning technique that seeks a direction $\mathbf{w}$ as a linear combination of the features over X so that the within cluster variances are minimized while at the same time the variances between cluster means and the total sample mean are maximized. Mathematically, its objective function is represented by:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}} \tag{4.2}$$

of which S_B and S_W respectively being called the between-cluster scatter matrix and within-cluster scatter matrix. They are calculated by:

$$S_B = \sum_k n_k (\mathbf{m}^{(k)} - \mathbf{m})(\mathbf{m}^{(k)} - \mathbf{m})^T \quad \text{and}$$
$$S_W = \sum_k \sum_i^{n_k} (\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})(\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})^T$$

with $\mathbf{m}^{(k)}$ and $\mathbf{m}$ are respectively the cluster mean of the k-th cluster and the total sample mean, and n_k is the number of instances grouped in k-th cluster. Solving this problem results in that $\mathbf{w}$ is the eigenvector corresponding to the largest eigenvalue of the matrix $S_W^{-1} S_B$. Usually in practice, when the number of clusters is only 2, one may need only a single dimension $\mathbf{w}$ to capture the solution. For a more general case where the number of clusters is $q+1$, a set of q eigenvectors $\mathbf{w}$'s corresponding to the q largest eigenvalues of $S_W^{-1} S_B$ are selected. Therefore in our approach, we choose such q eigenvectors as the optimal subspace S encoding for $C^{(1)}$. Each row in S corresponds to a projected data instance and the number of columns in S equals to the number of retained eigenvectors. Recall that the projected data instances in S strongly support the reference clustering $C^{(1)}$ (i.e., highly correlated to the cluster labels in $C^{(1)}$). Consequently, by taking the orthogonal condition of $S^T\mathbf{y}$, the newly mapped data, the y_i's, should be decorrelated from the reference solution $C^{(1)}$.

4.3 Solving the Constrained Objective Function. For solving the objective function with the constraint in Eq.(4.1), we can use the Lagrange method. First, let D be the diagonal matrix with $D_{ii} = \sum_j K_{ij}$ and let $L = D - K$, then by expanding the sum in Eq.(4.1), we can obtain the following function:

$$\begin{aligned}
&\sum_{i,j} \mathbf{f}^T \mathbf{x}_i K_{ij} \mathbf{x}_i^T \mathbf{f} - \sum_{i,j} \mathbf{f}^T \mathbf{x}_i K_{ij} \mathbf{x}_j^T \mathbf{f} \qquad (4.3)\\
&= \sum_i \mathbf{f}^T \mathbf{x}_i D_{ii} \mathbf{x}_i^T \mathbf{f} - \mathbf{f}^T X K X^T \mathbf{f}\\
&= \mathbf{f}^T X D X^T \mathbf{f} - \mathbf{f}^T X K X \mathbf{f}\\
&= \mathbf{f}^T X L X^T \mathbf{f}
\end{aligned}$$

We need to put another constraint $\mathbf{f}^T X D X^T \mathbf{f} = 1$ to remove $\mathbf{f}$'s scaling factor. Then using the Lagrange method with two Lagrange multipliers α and β, we solve the following function:

$$\mathcal{L}(\alpha, \beta, \mathbf{f}) = \mathbf{f}^T X L X^T \mathbf{f} - \alpha(\mathbf{f}^T X D X^T \mathbf{f} - 1) - \beta S^T X^T \mathbf{f} \tag{4.4}$$

For simplicity, let us denote:

$$\begin{cases} \widetilde{L} = XLX^T \\ \widetilde{D} = XDX^T \\ \widetilde{S} = XS \end{cases} \quad \text{and}$$

It is easy to verify that $\widetilde{D}$ is symmetric and positive semi-definite. Moreover, there exists $\widetilde{D}^{-1/2}$ and its transpose being identical. We therefore change the variable $\mathbf{f} = \widetilde{D}^{-1/2}\mathbf{z}$. It follows that:

$$\mathbf{f}^T \widetilde{L} \mathbf{f} = \mathbf{z}^T \widetilde{D}^{-1/2} \widetilde{L} \widetilde{D}^{-1/2} \mathbf{z} = \mathbf{z}^T Q \mathbf{z}$$

and the two constraints respectively are:

$$\mathbf{f}^T \widetilde{D} \mathbf{f} = \mathbf{z}^T \mathbf{z} = 1$$
$$\widetilde{S}^T \mathbf{f} = \widetilde{S}^T \widetilde{D}^{-1/2} \mathbf{z} = 0$$

Hence, our Lagrange function can be re-written as follows:

$$\mathcal{L}(\alpha, \beta, \mathbf{z}) = \frac{1}{2}\mathbf{z}^T Q \mathbf{z} - \frac{1}{2}\alpha(\mathbf{z}^T \mathbf{z} - 1) - \beta U^T \mathbf{z} \tag{4.5}$$

of which we have used U^T to denote $\widetilde{S}^T \widetilde{D}^{-1/2}$ and adding the constant $1/2$ does not affect our optimization objective. Taking the derivative of $\mathcal{L}(\alpha, \beta, \mathbf{z})$ with respect to $\mathbf{z}$ and setting it equal to zero give us:

$$\frac{\delta \mathcal{L}}{\delta \mathbf{z}} = Q\mathbf{z} - \alpha \mathbf{z} - \beta U = 0 \tag{4.6}$$

Left multiplying U^T to both sides results in $\beta = (U^T U)^{-1} U^T Q \mathbf{z}$ and substituting it into Eq.(4.6), we derive:

$$\begin{aligned}\alpha\mathbf{z} &= Q\mathbf{z} - U(U^TU)^{-1}U^TQ\mathbf{z}\\ &= \left(I - U(U^TU)^{-1}U^T\right)Q\mathbf{z}\\ &= PQ\mathbf{z}\end{aligned}$$

which is an eigenvalue problem with $P = I - U(U^TU)^{-1}U^T$. It is worth mentioning that PQ might not be symmetric albeit each of its individual matrices being symmetric. However, it is observed that $P^T = P$ and $P^2 = P$, so P is a projection matrix. Consequently, it is true that $\alpha(PQ) = \alpha(PQP)$ or equivalently the eigenvalues of both matrices PQ and PQP are the same. So instead of directly solving $PQ\mathbf{z} = \alpha\mathbf{z}$, we solve $PQP\mathbf{v} = \alpha\mathbf{v}$, with $\mathbf{v} = P^{-1}\mathbf{z}$.

Notice that the eigenvalues α_i's of PQP are always no less than zero and the smallest eigenvalue is indeed $\alpha_0 = 0$, corresponding to the eigenvector $\mathbf{v}_0 = P^{-1}\widetilde{D}^{1/2}\mathbf{1}$, where $\mathbf{1}$ is the unit vector. We thus remove such trivial eigenvalues/vectors from the solution. Consequently, the first nontrivial eigenvector $\mathbf{v}$ will correspond to the smallest non-zero eigenvalue α. This leads to our first optimum transformation vector:

$$\mathbf{f} = \widetilde{D}^{-1/2}P\mathbf{v}$$

and subsequently the optimal mapping feature $\mathbf{y}^T = \mathbf{f}^TX$. Generally, in the case where we want to use q transformation vectors to transform X into an q-dimensional subspace Y, i.e. $Y = F^TX$, we can select the set of q vectors $\mathbf{f} = \widetilde{D}^{-1/2}P\mathbf{v}$ corresponding to the q smallest positive eigenvalues of PQP. Similar to the FDA approach, we select q equal to the number of clusters desired for the alternative clustering $C^{(2)}$ minus 1. Given such novel mapping data, we apply the k-means to obtain the alternative clustering $C^{(2)}$.

It is worth mentioning that our algorithm can be extended to find multiple alternative clusterings based on the observation that it aims to find a subspace supporting each clustering solution. Therefore, it is straightforward to include all subspaces of previously found solutions as columns in the S matrix when searching for a novel alternative clustering. Certainly, the number of alternative clusterings can be given by the user or we can iterate the process until the total sum of square distances (computed in k-means) for a novel clustering is significantly larger than those of previously found clusterings.

5 Experiments.

In this section, we provide some initial experimental results regarding our method, which we name ACCP (Alternative Clustering with Constrained Projection) on synthetic and real-world datasets. Since our algorithm explores the subspace transformation approach, we mainly compare it against other techniques which also adopt this direction. In particular, its clustering performance is compared against two methods from [6] which we denote by Algo1, Algo2 respectively and the ADFT algorithm from [10]. For ADFT, we implement the gradient descent method integrated with the iterative projection technique (in learning the full family of the Mahalanobis distance matrix) [29, 30]. For all algorithms, including ours, we use k-means as the clustering algorithm applied in the transformed subspace.

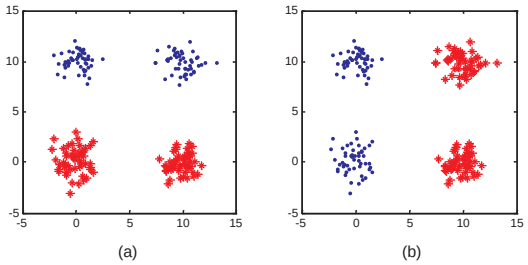


Figure 1: Alternative Clustering returned by our algorithm on the synthetic dataset (best visualization in color)

We evaluate the clustering results based on clustering dissimilarity and clustering quality measures. For measuring the dissimilarity/decorrelation between two clusterings, we use the normalized mutual information (NMI) that has been widely used in [11, 17, 27] and the Jaccard index (JI) which is used in [3, 10]. For measuring clustering quality, we use the Dunn Index (DI) [3, 10], which is defined by $DI(C) = \frac{\min_{i\neq j}\{\ \delta(c_i,c_j)\}}{\max_{1\leq\ell\leq k}\{\triangle(c_\ell)\}}$ where C is a clustering, $\delta:\ C\times C \to \mathbb{R}_0^+$ is the cluster-to-cluster distance and $\triangle: C\to\mathbb{R}_0^+$ is the cluster diameter measure. Note that for the NMI and JI measures, a smaller value is desirable, indicating higher dissimilarity between clusterings, whereas for the DI measure, a larger value is desirable, indicating a better clustering quality.

5.1 Experiments on Synthetic Data. For testing the performance of our algorithm on a synthetic dataset, we take a popular one from [10, 3] that is often used for alternative clusterings. This dataset consists of 4 Gaussian sub-classes, each containing 200 points in a 2-dimensional data space. The goal of using this synthetic dataset, when setting $k = 2$, is to test whether our algorithm can discover an alternative clustering that is

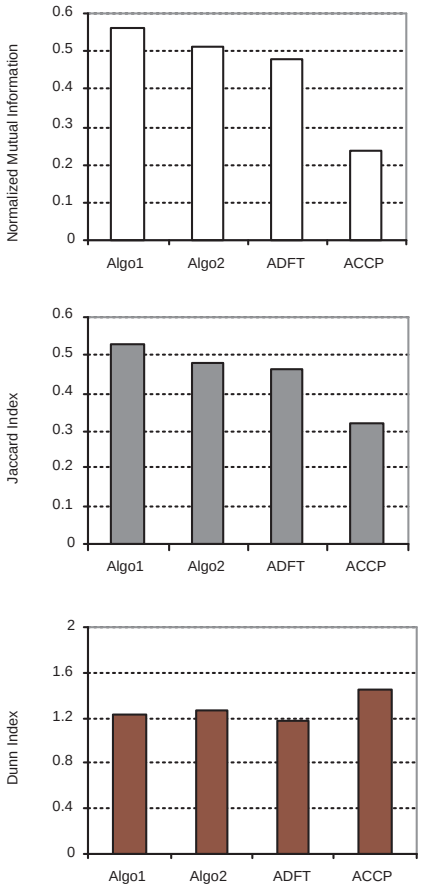


Figure 2: Clustering performance returned by four algorithms on the Cloud data

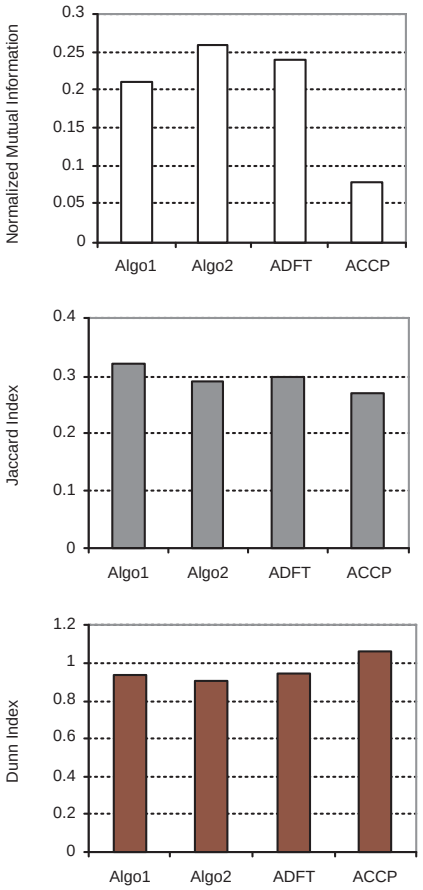


Figure 3: Clustering performance returned by four algorithms on the Housing data

orthogonal to the existing one. In Figure 1, we show the clustering results returned by our algorithm. Given the reference clustering $C^{(1)}$ which groups two Gaussians on the top and two Gaussians in the bottom as each cluster (shown in Figure 1(a)), ACCP successfully uncovers the alternative clustering $C^{(2)}$ which categorizes two Gaussians on the left and two ones on the right as clusters (shown in Figure 1(b)).

5.2 Experiments on the UCI data. We use two real-world datasets from the UCI KDD repository [2] to compare the performance of our algorithm against the other techniques. The first dataset is the Cloud data which consists of data collected from a cloud-seeding experiment in Tasmania in the time between mid-1964 and January 1971. For a reference clustering, we run the k-means algorithm (with $K = 4$ as the number of clusters) and its resultant clustering is used as the reference solution for all algorithms. The second dataset is the Housing data which consists information about the housing values collected in Boston's suburbs. Similar to the Cloud data, we apply k-means algorithm on this dataset and use it as the reference solution for all algorithms. We show the clustering performance of all techniques in Figures 2 and 3 respectively for the Cloud and Housing data.

As observed from all figures, our algorithm performs better than two techniques developed in [6] and the ADFT one in [10]. Its clustering dissimilarity measuring by the normalized mutual information and the Jaccard Index is lower than that of Algo1, Algo2 and ADFT whereas the clustering quality quantified by the Dunn Index is higher. This advantage can be justified by the approach of ACCP. Though all algorithms adopt a linear projection technique to derive a novel subspace that is decorrelated from the reference clustering, the

ACCP further takes into account the local neighborhood proximity of the data by retaining that property in the novel low dimensional subspace. The novel clustering structure in this learnt subspace therefore is more prominent compared to that of the other techniques which only attempt to minimize the correlation between two subspaces. Moreover, though the derived subspace of all techniques is ensured to be independent from the provided clustering, it is still possible that the alternative learnt from that subspace might not strongly decorrelated since there is a need to optimize for the clustering objective function as well. This generally explains for the difference in the NMI and JI measures of all methods.

6 Conclusions and Future Studies.

In this paper, we have addressed an important problem of uncovering multiple clustering views from a given dataset. We propose the ACCP framework to learn a novel lower dimensional subspace that is not only decorrelated from the provided reference clustering but the local geometrical proximity of the data is also being retained. By the second objective, our work goes beyond the others, which also adopt a subspace learning approach for seeking alternative clusterings, yet only focus on the dissimilarity property of the novel clustering. More importantly, we have demonstrated our dual-objective can be formulated via a constrained subspace learning problem of which a global optimum solution can be achieved.

Several initial empirical results of the proposed framework over the synthetic and real world datasets are given. Certainly, these results remain preliminary and more experimental work should be done in order to provide more insights into its performance. Additionally, though our framework is claimed to be capable of finding multiple clustering views, its performance on some suitable datasets has not been verified. Determining an ideal number of alternative clusterings still needs to be formally addressed. We leave these problems as an important and immediate task of the future work.

In addition to the above issues, our research in this paper opens up some potential and interesting directions for future studies. Specifically, despite advancing the subspace based learning approach for the alternative clustering problem, our research has exploited a hard constraint solution to ensure alternatives' dissimilarity. Such an approach might be too strict in some practical applications and thus some novel approaches based on soft constraints could be interesting. This implies that a trade-off factor between two criteria of subspaces' independence and intrinsic data structure retaining can be introduced, or we can constrain the subspace's decorrelation criteria to no less than a given threshold. In either of the two cases, a data-adaptive learning technique is required and the compromise between two factors is worth to study both theoretically and empirically. Furthermore, our research has not yet focused on the interpretation regarding the resultant alternative clusterings. This is an important problem, especially from the user perspective. Which features are best to describe a clustering solution and which ones are least correlated with respect to that solution are all informative to the user in understanding the data. Compared to a nonlinear subspace learning approach [7], our research direction in this work is beneficial due to its linear subspace learning approach. That means the linear combination amongst the original features is explicitly represented in the transformation matrix F. However, seeking for an optimal and concise set of features that best characterizes for a clustering solution is still a challenging issue given the fact that the number of original data features is usually huge. We believe that these open issues are worth to be further explored and studied.

References

[1] I. Assent, E. Müller, S. Günnemann, R. Krieger, and T. Seidl. Less is more: Non-redundant subspace clustering. In *1st MultiClust International Workshop in conjunction with 16th ACM SIGKDD.*, 2010.

[2] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[3] E. Bae and J. Bailey. COALA: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *The IEEE International Conference on Data Mining.*, pages 53–62, 2006.

[4] G. Chechik and N. Tishby. Extracting relevant structures with side information. In *International Conference on Neural Information Processing Systems (NIPS)*, 2002.

[5] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, August 1991.

[6] Y. Cui, X. Fern, and J. Dy. Non-redundant multi-view clustering via orthogonalization. In *The IEEE International Conference on Data Mining*, pages 133–142, 2007.

[7] X.H. Dang and J. Bailey. Generating multiple alternative clusterings via globally optimal subspaces. Under Review.

[8] X.H. Dang and J. Bailey. Generation of alternative clusterings using the cami approach. In *SIAM International Conference on Data Mining (SDM)*, pages 118–129, 2010.

[9] X.H. Dang and J. Bailey. A hierarchical information theoretic technique for the discovery of non linear alternative clusterings. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 573–582, 2010.

[10] I. Davidson and Z. Qi. Finding alternative clusterings using constraints. In *The IEEE International Conference on Data Mining*, pages 773–778, 2008.

[11] X. Fern and W. Lin. Cluster ensemble selection. *Stat. Anal. Data Ming*, 1(3):128–141, 2008.

[12] X. He and P. Niyogi. Locality preserving projections. In *International Conference on Neural Information Processing Systems (NIPS)*, 2003.

[13] P. Jain, R. Meka, and I. Dhillon. Simultaneous unsupervised learning of disparate clusterings. In *SIAM International Conference on Data Mining (SDM)*, pages 858–869, 2008.

[14] J. Kapur. *Measures of Information and their Application*. John Wiley, 1994.

[15] H.-P. Kriegel, E. Schubert, and A. Zimek. Evaluation of multiple clustering solutions. In *2nd MultiClust International Workshop in conjunction with ECML/PKDD*, 2011.

[16] H.-P. Kriegel and A. Zimek. Subspace clustering, ensemble clustering, alternative clustering, multiview clustering: What can we learn from each other. In *1st MultiClust International Workshop in conjunction with 16th ACM SIGKDD*, 2010.

[17] M. Law, A. Topchy, and A. Jain. Multiobjective data clustering. In *CVPR Conference*, pages 424–430, 2004.

[18] B. Mikhail and N. Partha. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 585–591, 2001.

[19] E. Müller, S. Günnemann, I. Färber, and T. Seidl. Discovering multiple clustering solutions: Grouping objects in different views of the data (tutorial). In *SIAM International Conference on Data Mining (SDM)*, 2011.

[20] Emmanuel Müller, Ira Assent, Stephan Günnemann, Patrick Gerwert, Matthias Hannen, Timm Jansen, and Thomas Seidl. A framework for evaluation and exploration of clustering algorithms in subspaces of high dimensional databases. In *BTW*, pages 347–366, 2011.

[21] X. V. Nguyen and J. Epps. minCEntropy: a novel information theoretic approach for the generation of alternative clusterings. In *The IEEE International Conference on Data Mining*, pages 521–530, 2010.

[22] D. Niu, J. G. Dy, and M. I. Jordan. Multiple non-redundant spectral clustering views. In *ICML*, pages 831–838, 2010.

[23] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[24] J. Principe, D. Xu, and J. Fisher. *Information Theoretic Learning*. John Wiley & Sons, 2000.

[25] Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 717–726, 2009.

[26] T. R. Sam and K. S. Lawrence. Nonlinear dimensionality reduction by locally linear embedding. *Science Journal*, 290(5500):2323–2326, 2000.

[27] A. Topchy, A. Jain, and W. Punch. A mixture model for clustering ensembles. In *SDM Conference*, 2004.

[28] J. Vreeken and A. Zimek. When pattern met subspace cluster - a relationship story. In *2nd MultiClust International Workshop in conjunction with ECML/PKDD*, 2011.

[29] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 505–512, 2002.

[30] L. Yang and R. Jin. Distance metric learning: A comprehensive survey, 2006.

Explorative Multi-View Clustering Using Frequent-Groupings

Martin Hahmann, Markus Dumat, Dirk Habich, Wolfgang Lehner
Dresden University of Technology
Database Technology Group
Dresden Germany
firstname.lastname@tu-dresden.de

Abstract

In this paper, we present our novel clustering technique called *frequent-groupings* as combination of alternative and ensemble clustering. Our new approach combines the benefit of both underlying concepts in an integrated way and allows the creation of robust clustering alternatives. In detail, we present (1) different approaches for the automatic creation of alternatives, (2) a user-guided exploratory extraction process and (3) an evaluation method to compare extracted alternatives with regards to their similarities.

1 Introduction

The relevance of clustering as a data analysis technique is more and more increasing. This is due to the ongoing trend for data gathering, that spreads in different areas of research and industry. In general, data clustering is described as the problem of (semi)-automatic partitioning of a set of objects into groups, called clusters, so that objects in the same cluster are similar, while objects in different clusters are dissimilar. The identification of similarities between data objects and their arrangement into groups with regard to these similarities, are essential tools to gain understanding and acquire previously unknown knowledge.

As this technique is quite powerful, its application by humans offers several challenges that need to be tackled, because they have a significant influence on the clustering result. One challenge e.g. is the selection of the best-fitting clustering algorithm for the dataset. A multitude of clustering algorithms has been developed over the years. These traditional algorithms have several limitations. On the one hand, they are not generally applicable, meaning that certain algorithms and parameterizations only suit certain datasets. On the other hand, traditional techniques generate only a single clustering result, but as todays data sets become more complex and high-dimensional there often are multiple valid clustering results possible for a single dataset. Besides these data centric problems, usability and applicability have become important issues as clustering evolves from a niche application in research to a widespread analysis technique employed in an increasing number of areas. With this trend, more users come into contact with clustering, who are often experts of their respective application domain but have no experience in the area of clustering. This calls for clustering approaches that can be versatilely applied and lack the complicated algorithm selection and configuration of traditional approaches.

In recent years, a number of approaches have been introduced to tackle some of these issues. The area of *alternative clustering* [2, 3] provides multiple clustering solutions for a dataset. With this, several views on complex data can be offered, and the very availability of multiple clusterings usually can free the user from adjusting a single clustering that proves unsatisfactory. An opposing approach is *ensemble clustering* [4, 9] in which a set of multiple clusterings is integrated to form a single consensus solution. This input set is also called clustering-ensemble and contains results that are generated using different algorithms and parameters. The consensus result is often more robust than clusterings generated by a single algorithm and set of parameters, which means that this technique is more versatile in terms of application. Additionally, algorithm selection and configuration is eased as a range of methods and parameters is utilized instead of a single configuration. On the downside, ensemble clustering provides just one solution and therefore resembles traditional clustering at that point. To create an alternative consensus clustering, the user has to re-configure the clustering-ensemble by changing algorithms and/or parameters. This is a very challenging task because multiple algorithms must be selected and configured.

The concepts of *alternative clustering* and *ensemble clustering* have their own benefits compared to traditional clustering approaches. However, to tackle all challenging issues of clustering from a users perspective, a combination of both concepts would be beneficial. In [6], we already introduced our hybrid concept

called *frequent-groupings*. This novel technique is based on the idea of frequent-itemset mining [1] and allows (1) the identification of robust clusters, occurring throughout the clustering-ensemble and (2) the derivation of robust alternative clustering results. After summarizing our concept of [6] in Section 2, we contribute the following detailed aspects to our hybrid concept in this paper:

1. We describe different automated approaches for the extraction of alternative clusterings in Section 3.

2. We propose a method for the explorative extraction of robust clustering alternatives and describe a way to measure their similarities in Section 4

Finally, we conclude the paper with some remarks regarding future work in Section 5 and a brief summary in Section 6.

2 Frequent Groupings

In our previous work [6], we described the concept of *frequent-groupings* as a means to combine benefits from the area of alternative clusterings and ensemble clustering. Using frequent-groupings, it is possible to combine clusters with an increased robustness into alternative clustering solutions.

The core idea of our approach originates from the area of ensemble-clustering [5], in which a single consensus solution is generated from an ensemble of different clusterings. Such a consensus solution is generated by identifying sets of objects, that are frequently assigned to the same cluster throughout the clustering-ensemble. These sets, then become the clusters of the consensus result and form a clustering, whose object assignments agree with the majority of the members of the clustering-ensemble.

Lets assume a dataset $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$ of objects in a multidimensional feature space, and a clustering-ensemble $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ that contains k clusterings of $\mathcal{D}$, each with a different number of clusters and different cluster composition. In general, there are two ways to evaluate the similarities of the cluster assignments of an object throughout the ensemble: (i) label-based approaches that match clusters [9] and (ii) approaches based on counting pairwise-similarities and the co-occurence of object pairs [4]. Our frequent-groupings approach [6] offers an alternative to these two principles and is based on the concept of frequent-itemsets [1].

Assuming a set of n items $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ and a set of transactions $\mathcal{T} = \{t_1, \ldots, t_m\}$ of which each transaction has a unique id an contains a subset of $\mathcal{I}$, a set of items $\mathcal{X}$ is called frequent if its *support* exceeds a given threshold. The support of an itemset $\mathcal{X}$ is defined by the fraction of transactions of $\mathcal{T}$ that contain $\mathcal{X}$. At this point, the analogy to ensemble clustering should be obvious: while frequent-itemsets aim to identify items that co-occur in many transitions, ensemble clustering searches for objects occuring together in the majority of clusters. For our frequent-groupings approach, we map the concepts of $\mathcal{I}$, $\mathcal{T}$ and *support* to the domain of ensemble-clustering, whereas $\mathcal{I}$ corresponds to the dataset and $\mathcal{T}$ corresponds to the clusters of the ensemble. According to these mappings, the *support* of a set of data objects $\mathcal{X}$ shows the fraction of the clustering-ensemble, in which $\mathcal{X}$ is part of the same cluster. If $support(\mathcal{X})$ exceeds a certain threshold, we call $\mathcal{X}$ a *frequent-grouping*.

To illustrate our frequent-groupings idea, we regard a small example from our previous work [6], featuring a dataset $\mathcal{D} = \{x_1, x_2, \ldots, x_9\}$ and a clustering-ensemble $\mathcal{C}$ containing the following 4 clusterings:

- $C_1 = \{(x_1, x_2, x_3)\ (x_4, x_5, x_6)\ (x_7, x_8, x_9)\}$
- $C_2 = \{(x_1, x_2, x_3)\ (x_4, x_5, x_7, x_9)\ (x_6, x_8)\}$
- $C_3 = \{(x_2)\ (x_1, x_3, x_4, x_5)(x_6)\ (x_7, x_8, x_9)\}$
- $C_4 = \{(x_1, x_2, x_3)\ (x_4, x_5)\ (x_6, x_8)\ (x_7, x_9)\}$

For frequent-grouping generation, we specify a minimal support threshold of 50%, meaning that objects occuring together in at least 2 clusterings of the ensemble are considered as frequent in our example. Figure 1 shows the frequent-groupings generated from the described setting.

The generation of frequent-groupings does not result in a simple list of object sets but in a more complex graph-like structure. This structure consists of nodes representing the frequent-groupings and edges showing subset/superset relations between frequent-groupings. Each node contains three values: (i) an id , (ii) a number showing the support of the grouping, and (iii) the objects that are part of this grouping. Besides the nine nodes with an id, there are several greyed-out nodes in the figure. These represent groupings that occur in the clustering-ensemble but do not meet the required minimal support and are thus filtered out. For example, each of three nodes in the upper part of Figure 1 occurs only in one of the clusterings of the ensemble. In order to further limit the number of actual frequent-groupings, we require that all frequent-groupings are *closed*, which means that no frequent-grouping fgX is a subset of another frequent-grouping fgY that has the same support as fgX. In compliance with this *closed* constraint the remaining greyed-out nodes at the base of the graph are rejected as valid frequent-groupings. As already mentioned, the edges of the graph show relations between

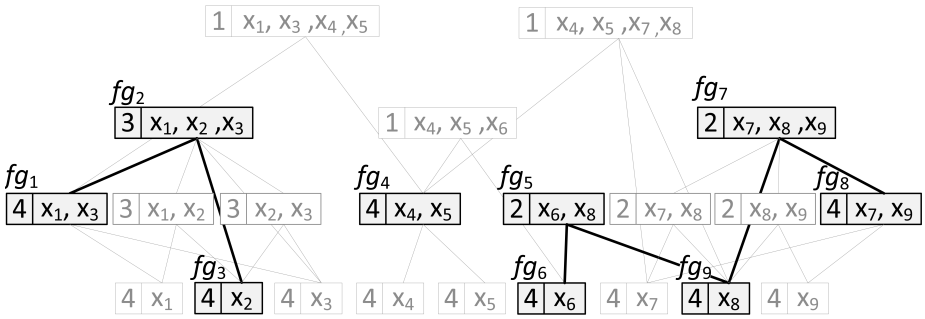


Figure 1: Example structure of frequent-groupings from [6].

the different frequent-groupings. Each edge represents a subset/superset relationship between the nodes it connects and thus plays a vital part in the construction of clustering alternatives.

To construct a clustering solution from the obtained frequent-groupings, we interpret them as clusters, and combine them in a way that all objects of $\mathcal{D}$ are part of one cluster. Regarding Figure 1, we can construct a clustering solution C_1, containing three clusters fg_2 ,fg_4 and fg_7 that include all objects of the dataset. As the generated frequent-groupings overlap, it is possible to produce multiple *alternative* combinations for the dataset. For example fg_2 has two subsets fg_1 and fg_3 that can replace fg_2, thus a first clustering alternative to C_1 can be created. Our small example shown in Figure 1 contains a total of six alternative consensus clusterings, which can be simply created by the user without any algorithmic assistance. Although manual creation of clustering alternatives is a valid option in the toy scenario described so far, it certainly is not for datasets and clustering-ensembles of a bigger scale. In the following sections, we introduce a scaled up scenario and apply our frequent-groupings approach in two different ways.

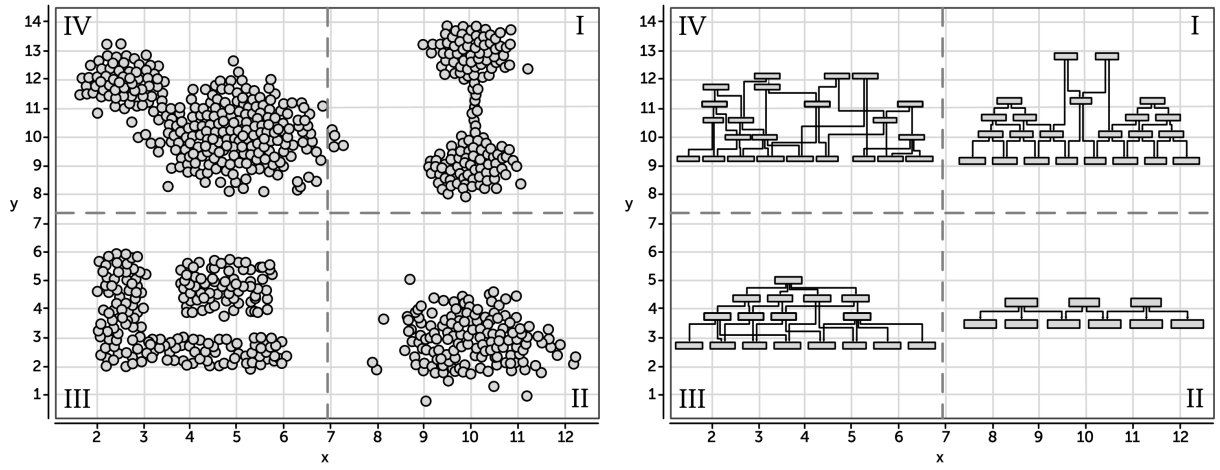
3 Automatic Extraction of Alternatives

As a running example for the remainder of this paper, we utilize the dataset depicted in Figure 2. The synthetic dataset contains about 1500 objects located in a 2-dimensional feature space, whereas the roman literal at each corner marks the respective quadrant of the dataspace. Although this can still be considered as a small setting, its scale is large enough to prevent manual processing as in the introductory example but still small enough to be handled in the scope of this paper. For our clustering example, we generated 10 different clusterings, using the *k-means* clustering algorithm with $k = 2, 3, 4, 5, 6, 7, 8, 9, 10, 15$ and a different initialization for each run. The dataset was crafted in order to provide structures that are hard to identify for partitioning algorithms like *k-means* e.g., the non-spherical clusters in quadrant III.

We already mentioned, the parallels of our frequent-groupings approach and the generation of frequent-itemsets in Section 2. Therefore, we examined existing algorithms for frequent-itemset mining in order to identify an efficient method for generating frequent-groupings. Regarding our scenario, we can state that the size of $\mathcal{I}$—1500 objects—is considerably higher than the size of $\mathcal{T}$, which contains only 69 clusters, found in our clustering-ensemble. We assume that in the most clustering scenarios, the number of objects will be higher than the number of clusters, and thus chose to employ the *CARPENTER* algorithm[8] for the extraction of frequent-groupings, as it is optimized for such a setting. CARPENTER works by enumerating transactions and intersecting them. It also utilizes different pruning techniques to optimize its runtime.

In order to use CARPENTER for the computation of frequent-groupings, we extended the algorithm so that the particular support value is stored with each frequent-grouping. For our running example, we applyed this method using a minimum support of 50%. As a result, we obtained a set of 72 frequent-groupings.

After obtaining the frequent-groupings, we want to generate multiple clustering alternatives from them. This is done by interpreting the frequent-groupings as clusters and combining them to a clustering where each element of the dataset is contained in exactly one frequent-grouping/cluster. To phrase it in another way: in a valid clustering alternative all clusters are disjoint and their union contains the complete underlying dataset. While other definitions of clustering alternatives are possible—e.g. objects are assigned to multiple clusters—we keep the former for the rest of this paper. The decision problem of combining our frequent-groupings in a way that the given conditions are satisfied, is also known as the 'exact cover problem' and



(a) Scatterplot of the data.

(b) Connected components / stem clusters per quadrant.

Figure 2: Running example for our scenario.

belongs to the class of NP-complete problems.

As a method to find all solutions of an exact cover problem, Donald E. Knuth proposed *Algorithm X*[7]. This algorithm works with an incidence matrix, representing the specific problem of exact cover and uses a recursive, depth-first search with backtracking to find all possible solutions. Knuth also proposes a concept for the efficient implementation of Algorithm X called *Dancing Links (DLX)*[7], which we used to find all clustering alternative.

When we applied DLX to our running example, the number of the resulting alternative clusterings turns out to be very high. With the input of all 72 frequent-groupings, the algorithm produces about 570.000 different valid clustering alternatives. It is safe to assume that the bulk of generated alternatives will be very similar. The reason for this is, that the minimum differences between clustering alternatives depend on the smallest frequent-groupings and very small groupings commonly occur in every clustering-ensemble. That means, a high number of very small frequent-groupings not only inflates the number of possible clustering alternatives, but also leads to results that only differ in the assignment of one or two objects. We are now faced with the challenge to find interesting clustering alternatives inside this vast amount of clusterings. In order to achieve this, we have to reduce the number of generated alternatives and find a way to identify distinctive solutions. In the following, we describe three unguided/automatic approaches to the issue.

Filtering A straightforward approach to this problem is to filter out very small frequent-groupings before the construction of alternatives. On the one hand, this reduces the the number of possible clusterings. On the other hand, the generated alternatives become more dissimilar and thus more interesting as the minimal frequent-grouping size is increased. Unfortunately, the filtering approach is troublesome work as an optimal threshold for the minimum size must be determined. Furthermore, each removal of a frequent grouping can lead to a situation where it is impossible to cover each element of the dataset with a frequent-grouping. Therefore, continuous tests for coverage are necessary during the procedure.

Scoring Another way for finding interesting and dissimilar alternatives would be to apply quality or similarity measures to the generated alternatives. In doing so, all obtained solutions could be ranked by score and presented to the user as a top-k list. For this approach, one or more appropriate measures for clustering-similarity and/or clustering-quality must be selected. Which is a very challenging task, regarding the number of available metrics and their applicability in different settings. Due to the high number of clusterings, the calculation of these measures needs significant computation effort. Furthermore, the top-k ranking requires the identification of an optimal k.

Greedy Top-k Selection This approach is similar to the described *Filtering*, as we try to limit the number of clustering alternatives by reducing the number of frequent-groupings. Instead of removing the smallest groupings, we sorted all frequent-groupings according to their two essential characteristics—size and support—in descending order. From this list, we choose the *top-k* biggest frequent-groupings as input for the generation of clustering alternatives. If the value for k is chosen randomly, it cannot be guaranteed that the selected groupings will cover the whole dataset, thus it is not

possible to guarantee that even one clustering can be constructed. In order to make sure that at least one valid result is generated, we used a kind of brute-force approach, i.e. we start with $k = 1$ and execute our DLX implementation. If no result is returned k is incremented and the algorithm is run again. This is repeated until at least one clustering is returned. Although this approach is not very elegant, it can narrow the generated alternatives down to a number in the single digits. However, after continued testing we found out that the effectiveness of this method is unpredictable and depends highly on the underlying clustering-ensemble. We applied the described procedure, using the dataset from our running example and different ensemble configurations. Some configurations returned 10 or less alternatives, while other configurations returned much larger result sets, containing up to 300.000 clustering alternatives.

None of the approaches is optimal to reach our goal of finding a compact set of interesting clustering alternatives. Therefore we introduce an additional approach in the following section, that reaches the described goal by including the user into the process of building robust clustering alternatives.

4 Explorative Extraction of Alternatives

In the previous section, we showed that the straightforward automatic extraction of clustering alternatives leaves the user with a very large set of results, of which most will be very similar. To tackle this issue, we described some approaches that aim to create not all, but a compact set of distinctive clustering alternatives. All of these automatic approaches for extraction have drawbacks, mostly due to configuration challenges and predictability. As full automation of the extraction seems to be suboptimal, we incorporate the user into the generation process, thus allowing an explorative creation of clustering alternatives.

The core idea of this kind of extraction is to use the graph structure of our frequent-groupings as guidance for the creation of clustering alternatives. Lets look at the graph of frequent-groupings of our running example. After generating all 72 frequent-groupings with a minimal support of 50%, this graph consists of the 4 connected components shown in Figure 2(b), of which each represents one quadrant of our dataspace. This one-to-one correspondence is random and results from our clustering-ensemble. Different ensemble configurations will also lead to different connected components. These connected components form the starting point of our explorative extraction and we will refer to them as *stem clusters*. All *stem clusters* are disjoint and their union contains all objects of the dataset. Therefore we could derive a clustering solution with 4 clusters from this setting. However, such a clustering would not be a valid result. It would not correctly represent the occurrence of clusters in the clustering ensemble, but just unites a set of overlapping frequent groupings. Therefore, each *stem cluster* must be differentiated into one of diverse final clusters—like *stem cells*—in order to create a valid clustering alternative. This differentiation works according to the frequent-grouping relations that are contained in the graph structure.

To illustrate this approach, let us regard the stem cluster/connected component for quadrant IV of our running example, that is depicted in Figure 3. The display is similar to the graph we already described in Figure 1, but due to its larger size and complexity some small changes were made. Again, each node represents a frequent-grouping and shows its information according to the pattern $id : [support] : size$. As the groupings feature a bigger size, we do not list all of their member identifiers but only display the size of the grouping. For better readability, we placed the nodes according to their support. At the top of the graph, we find groupings occurring in 5 of 10 clusterings—thus representing the minimal support requirement—while groupings that occurred in all clusterings of the ensemble are located at the lowest tier of the graph structure. Again, edges indicate subset/superset relations.

As stated, a frequent-grouping can be substituted by its subsets to generate a clustering alternative. These subset relations are used in the differentiation of the stem cluster. Based on this, we choose those nodes of the connected component that have no superset, as a starting points for the differentiation. The more subsets such a node has, the more possible alternatives can be created. As nodes with no supersets are generally few and contain a high number of objects, we can assume that the generated clustering alternatives will feature a small number of big clusters, which should be beneficial for the creation of dissimilar alternatives. For the remainder of this section, we will refer to these nodes as *roots*. The stem cluster/connected component in Figure 3 has 5 *roots*, namely *fg71, fg70, fg48, fg 65* and *fg55*. Each of these roots will be the starting point for one run of the differentiation algorithm described in the following.

Our proposed algorithm traverses the frequent-groupings graph of a stem cluster in a top-down fashion starting at one of its roots. Each run of our algorithm creates one clustering alternative, which means that the number of roots determines the maximum number of alternative clusterings that can be differentiated for a stem cluster. Based on this, two cases can occur during differentiation of a stem cluster. If there

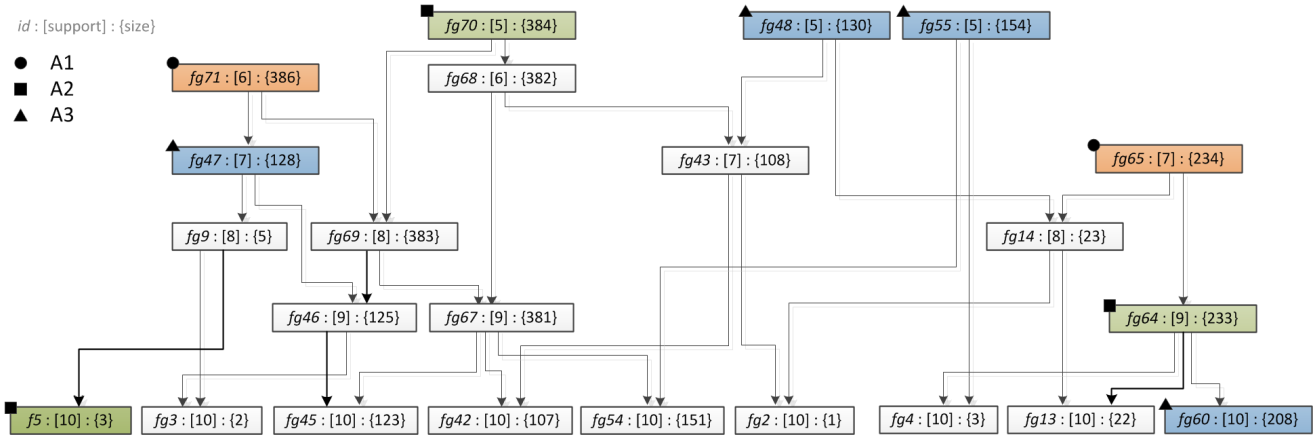


Figure 3: Frequent-groupings and clustering alternatives for quadrant 4 of the running example.

is exactly one root—first case—, then this root is the single most general alternative for the respective stem cluster. Therefore, it becomes a cluster in the clustering alternative and the differentiation is finished. In contrast—second case—, if there are multiple roots, the algorithm is executed for each root. Our proposed algorithm works as follows:

1. Select a root.(In the first iteration the root is provided. In further iterations the root with the highest size is selected.)
2. The selected root becomes a cluster for the clustering alternative.
3. Mark the successor set of the selected root.
4. Delete all nodes from the graph that are not part of the current root's successor set but have a path into it. This is done to remove all remaining groupings that overlap with the selected cluster.
5. Delete the current root and its successor set.
6. If the graph contains no more nodes, the algorithm terminates and outputs the resulting clustering alternative. Otherwise the algorithm is repeated from step one.

To illustrate the workflow of our algorithm, lets look at Figure 4. We execute our algorithm with the first of the four roots *fg71*. The root is selected and becomes the first cluster of our clustering alternative *A1*. The successor set of *fg71* is marked, which is depicted by a frame in Figure 4. Nodes are deleted according to step 4 and marked by a cross in the figure. After the successor set of *fg71* is deleted only one connected component remains. The single root of this connected component is *fg65* which is selected for the second iteration of our algorithm. After this iteration, the graph contains no further nodes, thus our algorithm terminates and outputs the clustering alternative *A1* containing the two clusters *fg71* and *fg65*. This alternative is also generated when starting with root *fg65*. The executions of our algorithm for the root *fg70* results in clustering alternative $A2 = \{fg70, fg64, fg5\}$, while the algorithm runs for roots *fg48* and *fg55* yield identical results which we summarize as clustering alternative $A3 = \{fg48, fg55, fg60, fg47\}$. This shows the predictability of our algorithms, from the 5 roots of the stem cluster, 5 clustering alternatives are created of which 3 are unique. The nodes of the three unique alternatives are shown in Figure 3 and marked with different symbols. After the differentiation is done, the user is presented with a compact list of alternatives for each stem cluster. The user then generates a complete clustering alternative that covers the whole dataset by selecting one alternative for each stem cluster. If the user wants more alternatives for a certain stem cluster, the clusters of its alternative differentiation can be turned into further stem clusters. As an example, regard $A1 = \{fg71, fg65\}$. The single-root connected component of *fg71* and its successor set—depicted in Figure 4—can be turned into a multi-root stem cluster by removing *fg71*. The new stem cluster has the roots *fg69* and *fg47*, thus offering at most 2 further clustering alternatives for the objects formerly covered by *fg71*.

Let us regard the alternatives for the stem cluster of quadrant IV in detail. They are depicted as scatter plots in Figure 6. As we assumed the generated alternatives in general contain a small number of big clusters. Nonetheless, the clusterings *A1* and *A2* are very similar, in fact they only disagree in the assignment

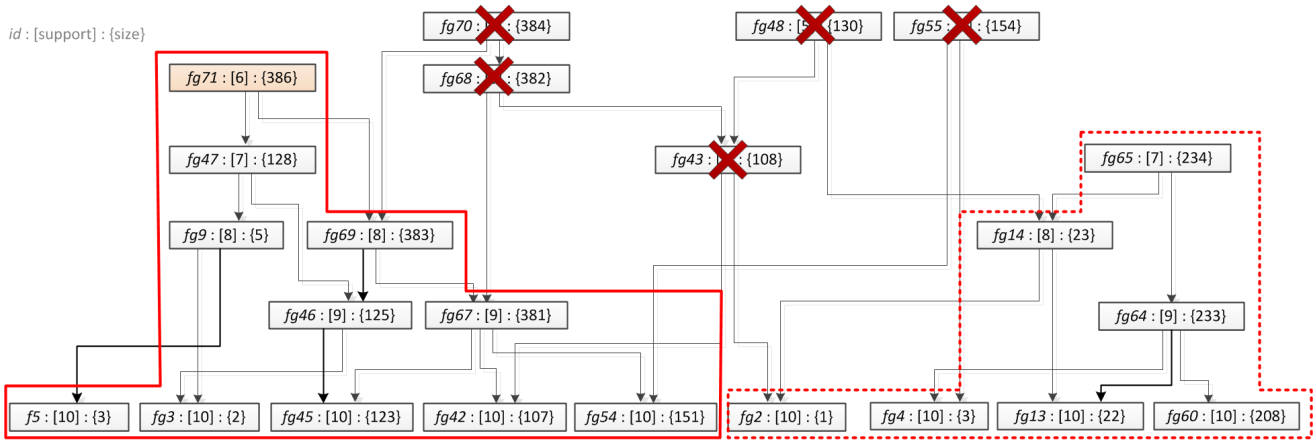


Figure 4: Execution of our algorithm with starting point *fg71*.

|fg71 ∩ fg48|	0	|fg48 ∩ fg71|	0
|fg71 ∩ fg55|	151	|fg48 ∩ fg65|	151
|fg71 ∩ fg47|	128	|fg55 ∩ fg71|	128
|fg71 ∩ fg60|	0	|fg55 ∩ fg65|	0
|fg65 ∩ fg48|	23	|fg47 ∩ fg71|	23
|fg65 ∩ fg55|	3	|fg47 ∩ fg65|	3
|fg65 ∩ fg47|	0	|fg60 ∩ fg71|	0
|fg65 ∩ fg60|	208	|fg60 ∩ fg65|	208

Figure 5: Maximum cluster intersection of *A1, A3*.

of 3 objects (*fg5*). So emphasizing the use of big clusters during the generation of alternatives does not necessarily lead to dissimilar alternatives. Therefore, in the following, we describe a similarity measure for our generated clustering alternatives, that again uses information from the frequent-groupings graph.

The idea for our similarity measure originates from the area of ensemble clustering. As already mentioned the goal of this technique is to identify sets of objects that are assigned to the same clusters in different clusterings. We transfer this idea to our scenario by using the number of objects that are placed in the same cluster in two different clustering alternatives as a measure of similarity between the two. In order to calculate this similarity, we intersect the clusters of both and look for the biggest sets of shared objects. Let us regard the alternatives *A1* and *A3* of our running example. All information regarding intersections can be easily derived from our frequent-grouping graph by examining subset/superset relations and mutual successors. We start by intersecting all clusters of *A1* with those of *A3*. The results of this intersection are shown on the left side of Figure 5. On the right side are the results of the intersections between *A3* and *A1*.

To get the overall intersection of both clusterings, we select the maximum intersection of each cluster of one alternative with the other alternative and sum them up. As we can see there occurs a problem that results from the different number of clusters in both alternatives. When we sum up the maximum intersections of the clusters of *A1* with *A3* we get a total intersection of $151 + 208 = 359$ objects. In contrast, if we add up the maximum intersections of the clusters of *A3* with *A1* we get an overall intersection of $23 + 151 + 128 + 208 = 510$ objects, which is higher than the actual object-count of the respective stem cluster. This means our similarity measure is not commutative. This characteristic comes from the subset/superset relations we use to derive the maximum intersections. In order to get the correct overlap we select the maxima from *A1* with *A3* and *A3* with *A1* intersect both result sets. In doing so we only get those cluster overlaps that exist in both directions, thus resulting in $151 + 208 = 359$. As the stem cluster for quadrant 4 covers 425 objects of the dataset, the similarity of *A1* and *A3* is 84%. When we calculate the similarity of *A1* and *A2* we get a high similarity of 99% which we already observed in Figure 6. This similarity information can be used by the user during the selection of alternatives for stem clusters.

5 Future Work

With our proposed extraction approach a user explores the frequent-groupings graph in a top-down fashion and generates different clustering alternatives on the way. As these alternatives are generated according to the subsets found under each root node, we can generally state that alternatives are created by splitting up stem clusters in different ways. This *split* is one of the four high-level feedback operations we already introduced in [5], that can be used to shape a clustering. A major

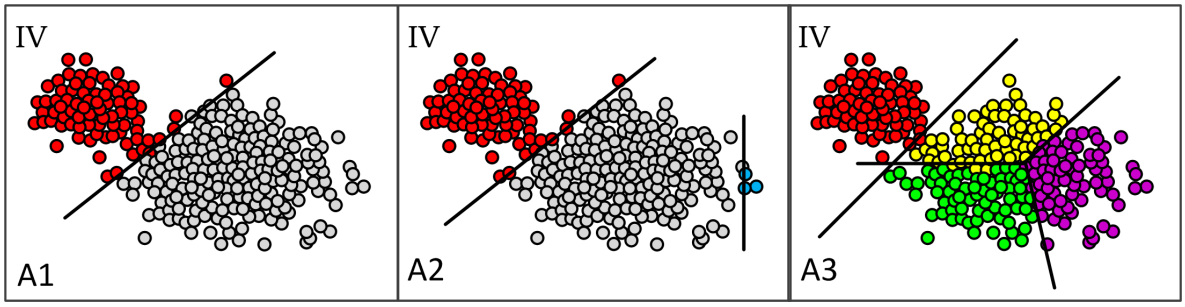


Figure 6: Scatter plots of clustering alternatives *A1, A2* and *A3*.

part of our future work will be, to map the remaining operations *merge, refine* and *restructure* to our frequent groupings scenario.

As *split* works in a top-down fashion its contrary operation *merge* would resemble a bottom-up procedure. In such a scenario the starting point for our exploratory extraction, would be created from the smaller high support frequent-groupings at the bottom of the graph. By traversing the ascending edges of our graph, these smaller clusters/frequent-groupings can be combined into larger clusters. Thus new alternative clusterings could be generated. An exploratory extraction based on the *merge* operation would also require a new equivalent for the stem clusters we introduced in this paper.

In general we will examine new ways to determine a suitable starting point for the exploratory extraction. As pure top-down or bottom-up approaches start at only one end of the frequent-groupings graph—root or leaf—and just allow one direction of traversal, it would be interesting to look for a starting point in between those extremes. Such a median start would allow an exploration using *split* and *merge* in combination. For this kind of scenario it is also important to look at the influence of different minimal supports on the graph structure.

6 Conclusion

In this paper, we expanded our frequent-grouping concept [6] for the generation of robust clustering alternatives. We described the generation of our frequent-groupings and illustrated ways for the automatic extraction of all possible clustering alternatives. After that we pointed out the challenge of selecting interesting and dissimilar clustering alternatives from the vast amount of solutions, that can be generated using automatic extraction. To tackle this challenge, we proposed an exploratory approach, that allows the predictable creation of a manageable number of alternative clustering solutions. In addition, we derived a similarity measure for our clustering alternatives, that can aid the user during the extraction process.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. pages 487–499, 1994.

[2] E. Bae and J. Bailey. Coala: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006)*, pages 53–62, 2006.

[3] X. H. Dang and J. Bailey. Generation of alternative clusterings using the cami approach. In *Proceedings of the Tenth SIAM International Conference on Data Mining*, pages 118–129, 2010.

[4] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. In *Proc. of ICDE*, 2005.

[5] M. Hahmann, D. Habich, and W. Lehner. Evolving ensemble-clustering to a feedback-driven process. In *Proceedings of the IEEE ICDM Workshop on Visual Analytics and Knowledge Discovery (VAKD)*, 2010.

[6] M. Hahmann, D. Habich, and W. Lehner. Browsing robust clustering-alternatives. In *Proceedings of the 2nd MultiClust Workshop: Discovering, Summarizing and Using Multiple Clusterings (MultiClust 2011)*, 2011.

[7] D. E. Knuth. Dancing links. In *Millenial Perspectives in Computer Science: Proceedings of the 1999 Oxford-Microsoft Symposion in Honour of Sir Tony Hoare*, pages 187–214. Palgrave, 2000.

[8] F. Pan, G. Cong, A. K. H. Tung, J. Yang, and M. J. Zaki. Carpenter: finding closed patterns in long biological datasets. In *KDD*, pages 637–642, 2003.

[9] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3, 2002.