A Semi-Supervised Approach to Projected Clustering with Applications to Microarray Data

Kevin Y. Yip[1], Lin Cheung[2], David W. Cheung[2] Liping Jing[3], and Michael K. Ng[3]

[1]Yale University, Email: yuklap.yip@yale.edu
[2]The University of Hong Kong, Email: {lcheung,dcheung}@csis.hku.hk
[3]Hong Kong Baptist University, Email: lpjinghk@gmail.com, and mng@math.hkbu.edu.hk

Abstract: Recent studies have suggested that extremely low dimensional projected clusters (e.g. $< 10\%$ of the total number of dimensions) exist in real datasets such as gene expression profiles. A number of algorithms have been proposed to detect clusters in subspaces, but few can identify clusters with such low percentage of relevant dimensions. In this paper we propose a new algorithm that can accurately identify this kind of clusters. It uses a robust objective function to combine object clustering and dimension selection into a single optimization problem. It also allows the input of domain knowledge in various forms to improve the clustering accuracy. Both theoretical analyses and experimental results show that by using a small amount of input knowledge, possibly covering only a portion of the underlying classes, the new algorithm can precisely detect clusters with only 1% of the dimensions being relevant. We also show that the semi-supervised approach allows the algorithm to identify a particular target set of clusters when there are multiple meaningful groupings of the objects. A real mircoarray data set is used to demonstrate how to use input knowledge to improve the clustering accuracy results.

Keywords: Data mining, Mining methods and algorithms, Clustering.

Reference to this paper should be made as follows: Kevin Y. Yip, Lin Cheung, David W. Cheung, Liping Jing and Michael K. Ng (2007) 'A Semi-Supervised Approach to Projected Clustering with Applications to Microarray Data', *Int. J. Data Mining and Bioinformatics*, Vol. x, No. x, pp.xxx–xxx.

Biographical Notes: Kevin Y. Yip is with the Department of Computer Science, Yale University, New Haven, Connecticut, USA.

Lin Cheung and David W. Cheung are with the Department of Computer Science, The University of Hong Kong, Hong Kong.

Liping Jing and Michael K. Ng are with the Center for Mathematical Imaging and Vision, and Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong

1 Introduction

Recently many studies have suggested the presence of low dimensional clusters in high-dimensional real datasets. For example, in a typical microarray gene expression dataset that contains the expression values of several thousands of genes in different samples, it is common to find only several tens of genes having expression patterns that are highly specific to each cluster of samples(Pomeroy et al., 2002). The genes are called the relevant genes, as opposed to the irrelevant genes that do not help much in identifying the cluster members (i.e., samples of the same type). Due to the large number of genes being irrelevant to each cluster, two samples in the same cluster could have low similarity when measured by a similarity function that considers the expression values of all genes. The clusters may thus be undetectable by traditional clustering algorithms. The same kind of low dimensional clusters could also exist in datasets from various domains such as computer vision(Procopiuc et al., 2002), e-commerce(Wang et al., 2002), text mining and nutrition value analysis(Yip, 2003).

The projected clustering problem(Aggarwal et al., 1999) is defined for such a scenario. Each projected cluster is a set of objects with an associated set of relevant dimensions such that in the subspace formed by the relevant dimensions, the objects are similar to each other but dissimilar to objects outside the cluster. In this paper we measure object similarity based on Euclidean distance, so a dimension is more relevant to a cluster if the projections of its members on the dimension are closer to each other, but more remote from other objects. The goal of a projected clustering algorithm is to identify clusters of objects and their relevant dimensions such that a certain objective function (e.g. within-cluster dispersion along the relevant dimensions) is optimized.

While the clusters in real datasets could contain an extremely low percentage of relevant dimensions (e.g. less than 10% of all genes in a gene expression dataset), it has been reported that most current projected clustering algorithms are unable to identify clusters with such low dimensionality(Yip et al., 2004). This is mainly due to their use of objective functions that highly rely on the accuracy of some input parameters, and the use of similarity calculations that involve all dimensions, which may not reflect the real similarity between different objects(Yip et al., 2004).

In addition, being unsupervised methods, these clustering algorithms make little use of domain knowledge, even some domain knowledge is usually available in some applications. For example, in gene expression datasets, the functions of some genes are usually known to the biologists. In text mining, some document types have well-known keywords that help identify the member documents. In order to better utilize domain knowledge in the clustering process, a number of *semi-supervised* clustering algorithms have been proposed(Demiriz et al., 1999). For example, in a semi-supervised k-means algorithm(Wagstaff et al., 2001), domain knowledge about the relationships between some object pairs is used to force the assignment of some pairs to the same cluster and some to different clusters. As reported in many studies

(e.g.(Cohn et al., 2003; Wagstaff and Cardie, 2000)), the clustering accuracy can be greatly improved by inputting only a small amount of domain knowledge.

It should be noted that while some domain knowledge is being used by the algorithms, semi-supervised clustering is different from classification (supervised-learning) in that the knowledge being used in semi-supervised clustering may not be suitable or sufficient for classification. First, the knowledge needs not be in the form of class labels of the objects as required by classification. Second, the amount of knowledge for each class can be so small that it is insufficient for building a classifier that captures the general properties of the class. For example, the input knowledge can be biased towards one side of the class. But in such cases the input knowledge is still highly useful in semi-supervised clustering. Third, as opposed to classification, the input knowledge of semi-supervised clustering needs not cover all classes, but it is still possible to produce the clusters corresponding to every underlying class.

Previous studies on semi-supervised clustering do not consider the relevance of dimensions. But the semi-supervised approach is actually very useful in projected clustering. Given a small amount of example objects of a cluster (e.g. tumor samples known to be of a certain type), the relevant dimensions of the cluster can be estimated as the dimensions along which the objects are significantly close to each other. Similarly, having a dimension specified as relevant to a cluster (e.g. a gene known to be relevant to a tumor type), the cluster members can be estimated from regions with unexpectedly high object densities.

Semi-supervised clustering also has an important application in handling datasets that have multiple meaningful groupings. For example, in cancer study, patients can be grouped by their response to a certain treatment, or by the risk of having cancer recurrence. Unsupervised methods can only produce a single set of clusters, which may correspond to only one of the groupings, or even none of them. Using the semi-supervised approach, by supplying different input knowledge, a single clustering algorithm can be guided to produce both kinds of clusters in different runs. From a machine learning point of view, the search space of semi-supervised clustering is very large and contains many local optima. The input knowledge helps clustering algorithms start searching at a point close to the target optimum, and guides the search path towards it.

All the above observations motivate the current study, which has three major contributions:

- Proposing a robust objective function for projected clustering that naturally involves dimension selection in the optimization process.
- Proposing the use of various forms of domain knowledge to improve the accuracy of projected clustering.
- Developing a new clustering algorithm that can (theoretically and empirically) detect clusters of extremely low dimensionality even with no input domain knowledge, and whose accuracy can be further improved by incorporating some domain knowledge in the clustering process.

In the next section we will review some related work in projected clustering and semi-supervised clustering. In Section 3 we will formally define the semi-supervised projected clustering problem, as well as the assumptions being made in

this study. In Section 4 we will describe our new algorithm. Experimental results will be presented in Section 5, together with some observations and discussions. In Section 6 we will summarize the whole study and discuss some future extensions.

2 Related Work

2.1 Projected Clustering

Existing projected clustering algorithms can be classified into three categories: partitional, one cluster at a time, and hierarchical. The partitional approach PROCLUS(Aggarwal et al., 1999) is based on the traditional k-medoids approach(Ng and Han, 1994), with a goal of minimizing the average within-cluster dispersion. The distance between different cluster members is computed in the relevant subspace of the cluster, which is determined by measuring the average distance between the medoid and a set of "neighboring objects" that are close to it when all dimensions are considered. The dimensions with the smallest average distances to the medoid of each cluster are selected as the relevant dimensions of the cluster, which form its relevant subspace.

Another partitional method ORCLUS(Aggarwal and Yu, 2000) improves PROCLUS by selecting principal components so that clusters not parallel to the original dimensions can also be detected. It also adds a hierarchical part that can potentially reduce the errors due to inaccurate initial object assignments.

A limitation of these partitional methods is the determination of neighboring objects based on similarity calculations that involve all dimensions. Since different members of a cluster may appear to be dissimilar when all dimensions are considered, the neighboring objects of a medoid need not be come from the same real cluster and the relevant dimensions suggested by them could be wrong. Also, as the approaches use an objective function that tends to give better scores when fewer dimensions are regarded as relevant to a cluster(Yip et al., 2004; Yip, 2003), they require users to supply the average number of relevant dimensions per cluster, which is usually unknown to users. If improper values are used, the clustering accuracy can be seriously affected(Yip et al., 2004).

The Monte Carlo methods DOC and FastDOC(Procopiuc et al., 2002) identify projected clusters one after another. To find a cluster, an object is randomly selected as the seed, and some other objects are randomly sampled to determine the relevant subspace of the cluster. A dimension is regarded as relevant to the cluster if all the objects are within a distance ω from the seed along the dimension. Each cluster is thus a hypercube of width 2ω. The more objects and relevant dimensions a cluster has, the less likely it is formed by chance, and thus it receives a better score. The relative importance between the number of objects and relevant dimensions is controlled by a user parameter β. The algorithm repeatedly tries different seeds and neighboring objects and returns the cluster with the highest score. Then the whole process is repeated for a new cluster.

The algorithms perform well when each cluster is in the form of a hypercube and the parameter values are specified correctly, but in many cases these requirements cannot be met and the clustering results are quite unsatisfactory(Yip et al., 2004).

The number of seeds and neighboring objects required to try could also be so large that causes the algorithms to run for a long time.

The hierarchical algorithm HARP is proposed in(Yip et al., 2004). Its basic assumption is that two objects are likely to belong to the same cluster if they are very similar to each other along many dimensions. Clusters are allowed to merge only if they are similar enough in a number of dimensions, where the minimum similarity and minimum number of similar dimensions are controlled by two thresholds. At the beginning, the thresholds are set to some harsh values such that only merges that are very likely to group objects belonging to the same real cluster are allowed. As the relevant dimensions of each cluster becomes more apparent, the threshold values are loosened to allow more merges. The process repeats until the thresholds reach their baseline values, or a target number of clusters is reached.

The method successfully avoids extensive distance calculations that involve all dimensions, and user parameters whose values are hard to determine. However, due to the hierarchical nature, it is intrinsically slow. Also, if the number of relevant dimensions per cluster is extremely low (e.g. 5% of the dataset dimensionality), the accuracy of HARP may drop as the basic assumption becomes less valid due to the presence of large amount of noise values in the dataset.

Recently, the problem of finding projected clusters from streaming data has also been studied(Aggarwal et al., 2003).

In summary, most of the existing projected clustering algorithms make use of objective functions whose effectiveness rely greatly on the accuracy of some parameter values that are hard for users to determine. Some of them involve similarity calculations that consider all dimensions, which can be quite misleading when the cluster dimensionality is small. A thorough survey of the above algorithms and others proposed for two related problems, namely subspace clustering(Agrawal et al., 1998) and biclustering(Cheng and Church, 2000), can be found in(Yip, 2003).

2.2 Semi-supervised Clustering

A recent trend in machine learning research has been to combine the techniques developed for unsupervised learning and supervised learning to handle datasets with partial external information. One of the foci is semi-supervised clustering, which actively uses the available domain knowledge in guiding the clustering process. These methods can be categorized according to the kinds of knowledge being input, the time that the knowledge is input, and the way the knowledge is used to affect the clustering process.

The simplest type of input is labeled objects(Basu et al., 2002; Demiriz et al., 1999). In some cases, users do not know the exact class labels of objects, but they have some knowledge on which objects should be/should not be put into the same cluster, which can be specified by must-links and cannot-links(Basu et al., 2004; Bilenko et al., 2004; Klein et al., 2002; Wagstaff and Cardie, 2000; Wagstaff et al., 2001; Basu et al., 2004). Some other studies propose the input of classification rules(Talavera and Bejar, 1999), examples of similar objects(Xing et al., 2003), or even general comments such as which cluster a particular object should not be put into(Cohn et al., 2003).

The knowledge can be supplied at different time. It can be supplied before clustering to guide the clustering process(Basu et al., 2002, 2004; Bilenko et al.,

2004; Demiriz et al., 1999; Klein et al., 2002; Wagstaff and Cardie, 2000; Wagstaff et al., 2001; Xing et al., 2003; Basu et al., 2004), or after clustering to evaluate the clusters and guide the next round of clustering(Cohn et al., 2003). Some algorithms can also actively request users to supply some specific information at the most appropriate time(Basu et al., 2004; Klein et al., 2002).

There are various ways to use the input knowledge, such as guiding the formation of seed clusters(Basu et al., 2002, 2004; Bilenko et al., 2004; Basu et al., 2004), forcing or recommending some objects to be put into the same cluster or different clusters(Wagstaff and Cardie, 2000; Wagstaff et al., 2001), and modifying the objective function(Basu et al., 2004; Bilenko et al., 2004; Demiriz et al., 1999; Basu et al., 2004), similarity function(Cohn et al., 2003; Xing et al., 2003; Bilenko et al., 2004; Basu et al., 2004) or distance matrix(Klein et al., 2002).

A related problem is semi-supervised classification, which aims at using unlabeled data to build more accurate classifiers. See, for example(Ratsaby and Venkatesh, 1995; Blum and Mitchell, 1998; Szummer and Jaakkola, 2001) for details.

3 Problem Definition

In this section we formally define the semi-supervised projected clustering problem. We first describe the data model. The input dataset D contains n objects and d dimensions. Each object is either randomly sampled from one of the k underlying hidden classes, or is a random outlier. The i-th class is represented by a random vector c_i of d variables each corresponding to one of the d dimensions. Each class is associated with a set of relevant dimensions that form the relevant subspace of it. Denote c_{ij} as the random variable of c_i along the j-th dimension v_j, and σ_{ij}^2 as the variance of c_{ij}. If v_j is relevant to c_i, then c_{ij} follows a Gaussian distribution with a small σ_{ij}^2. If v_j is irrelevant to c_{ij}, then c_{ij} follows an unknown distribution with a large σ_{ij}^2.

In dataset D, the samples generated from the k classes can be grouped to form k clusters $\{C_i\}_{i=1}^k$ with the corresponding sample variance along dimension v_j as s_{ij}^2. Denote s_j^2 as the sample variance of all objects in D along v_j. Due to the underlying class model, s_{ij}^2 is expected to be much smaller than s_j^2 except in the very rare case that v_j is relevant to all classes and all the class centers projected onto the dimension are very close. In another view, given a cluster with an unknown relevant subspace, a dimension is more likely to be relevant to the cluster if its projection on the dimension has a smaller variance.

We will call the above clusters the "real clusters" since they are defined according to the actual hidden classes. For simplicity, we will simply call the clusters produced by a clustering algorithm the "clusters". A clustering algorithm determines the dimensions relevant to a cluster through the dimension selection process. We will call them the "selected dimensions" of a cluster.

The semi-supervised projected clustering problem is then defined as follows. The inputs to the problem are:

- The dataset D
- The target number of clusters k

- A set I^o of labeled objects (<obj. ID, class label> pairs), each states that the object is a sample of the class. The set may or may not cover all classes.
- A set I^v of labeled dimensions (<dim. ID, class label> pairs), each states that the dimension is relevant to the class. Each dimension can be specified as relevant to multiple classes. The set may or may not cover all classes.
- A set M^o of must-link object pairs, each states that the two objects are samples of the same class.
- A set C^o of cannot-link object pairs, each states that the two objects are samples of different classes.

Each of the sets I^o, I^v, M^o and C^o can be empty, which means users have the flexibility to input whatever kinds of knowledge available, or even not to input any knowledge at all.

The goal is to identify k clusters and their selected dimensions, and a (possibly empty) list of outliers, such that an objective function (to be described below) is optimized.

Before clustering, the knowledge being input is preprocessed as follows. Given a must-link object pair $\{x, y\} \in M^o$, if the label of either of the objects is given in I^o (e.g. (x, i)), the must-link pair will be removed from M^o and both objects will have an entry in I^o with the specified class ID as label, i.e., both (x, i) and (y, i) will be in I^o.

Then we allow users to choose whether to infer new knowledge from the inputs. The inference rules are:

1. $\{x, y\} \in M^o \wedge \{y, z\} \in M^o \Rightarrow \{x, z\} \in M^o$
2. $(x, i) \in I^o \wedge (y, i) \in I^o \Rightarrow \{x, y\} \in M^o$
3. $\{x, y\} \in M^o \wedge \{y, z\} \in C^o \Rightarrow \{x, z\} \in C^o$
4. $(x, i_1) \in I^o \wedge (y, i_2) \in I^o \wedge i_1 \neq i_2 \Rightarrow \{x, y\} \in C^o$.

If the inputs are highly reliable, it is preferable to perform the inference in order to give maximum guidance to the clustering process. On the other hand, if it is likely that some inputs are incorrect, performing the inference may result in fake knowledge that misleads the algorithm and lowers the clustering accuracy. If the user has good understanding of the inputs, he/she may choose to perform only a selected subset of the inferences.

We confine the scope of the current study by a number of assumptions, most of which are also implicitly or explicitly made in previous studies on projected clustering and semi-supervised clustering. The possibility of relaxing some of them will be discussed in Section 6.

1. The determination of relevant dimensions is mainly to help identify the object clusters (as opposed to biclustering(Cheng and Church, 2000) where both rows and columns are treated equally). In other words, the major goal of the algorithm is to form good clusters of objects, determining relevant dimension is simply an auxiliary way to achieve this goal.

2. Clusters are disjoint and axis-parallel (as opposed to subspace clustering(Agrawal et al., 1998) and ORCLUS(Aggarwal and Yu, 2000) respectively).
3. Object similarity is based on the difference between projected values (as opposed to pattern-based clustering(Pei et al., 2003; Wang et al., 2002)).
4. All objects of a class are close to each other in a certain subspace, so that one class corresponds to only one real cluster (as opposed to decision trees(Quinlan, 1993) where the objects of one class can form multiple clusters).
5. The knowledge inputs do not contradict with each other (e.g. two objects are simultaneously must-linked and cannot-linked), although some may be incorrect.

3.1 Objective Function

As discussed, a cluster of objects is likely to be from the same real cluster if their projections are unexpectedly close to each other along many dimensions. Intuitively, we need an objective function that captures the within-cluster dispersion along the relevant dimensions. We designed the following function ϕ for this purpose:

$$\phi = \frac{1}{nd}\sum_{i=1}^{k}\phi_i \tag{1}$$

$$\phi_i = \sum_{v_j \in V_i}\phi_{ij} \tag{2}$$

$$\phi_{ij} = n_i - 1 - \frac{1}{\hat{s}_{ij}^2}\sum_{x \in C_i}(x_j - \tilde{\mu}_{ij})^2 - \rho_{ij} \tag{3}$$

$$= (n_i - 1)(1 - \frac{s_{ij}^2 + (\mu_{ij} - \tilde{\mu}_{ij})^2}{\hat{s}_{ij}^2}) - \rho_{ij} \tag{4}$$

$$\begin{aligned}\rho_{ij} = \{&\omega_o \sum_{(x,i) \in I^o \wedge x \notin C_i}[1 - \frac{(x_j - \tilde{\mu}_{ij})^2}{\hat{s}_{ij}^2}] + \\ &\frac{\omega_m}{2}\sum_{\{x,y\} \in M^o \wedge x \in C_i \wedge y \notin C_i}[1 - \frac{(x_j - y_j)^2}{\hat{s}_{ij}^2}] + \\ &\omega_c \sum_{\{x,y\} \in C^o \wedge x \in C_i \wedge y \in C_i}\frac{(x_j - y_j)^2}{\hat{s}_{ij}^2}\},\end{aligned} \tag{5}$$

where n_i and V_i are the size and the set of selected dimensions of cluster C_i, x_j is the projection of an object x on dimension v_j, $\tilde{\mu}_{ij}$, μ_{ij} and s_{ij}^2 are the sample median, mean and variance of the projection of C_i on v_j respectively, and $\hat{s}_{ij}^2$ are the selection thresholds. The thresholds are used to normalize ϕ_{ij} to $[0, 1]$ such that the score components of different dimensions become comparable. They also play an important role in dimension selection (more on ϕ_{ij} will be discussed in the next section). The objective function ϕ is composed of the score components ϕ_i of each cluster, which in turn is the sum, over all selected dimensions, of the score

Figure 1 The dimension selection procedure.

```
Procedure SelectDim(C_i: target cluster)
1    Foreach dimension v_j do
2        Select v_j for C_i if and only if φ_ij > 0
End
```

components ϕ_{ij} that compute the within-cluster dispersion, deducted by the penalty ρ_{ij}. The ω's (ω_o, ω_m, ω_c) are confidence parameters for specifying how likely the input items are correct and the relative importance of different types of input. It is trivial to generalize the objective function such that each input has a separate weight, as in(Bilenko et al., 2004). But the large number of parameters usually overwhelms users and in many applications it is good enough to have only one confidence parameter per type. We thus adopt the simpler formulation. Overall a higher value of ϕ indicates a better set of clusters, therefore the goal of our clustering algorithm is to maximize the value of ϕ.

The magnitude of a penalty depends on how reasonable it is to violate the input. If a labeled object is assigned to another cluster, then the closer it is to the center of the input-specified cluster, the higher will be the penalty. This is because it is unreasonable to violate the input if assigning the object to the specified cluster leads to a large increase of the objective score. For must-link object pairs, if the two objects are put into different clusters, then the closer are the objects, the higher will be the penalty. The reverse holds for cannot-link pairs. A similar argument can be found in(Basu et al., 2003). Notice that in(Bilenko et al., 2004), there is a counter proposal in which if a must-link input is violated, then the penalty is higher if the two objects are farther apart. The rationale is that such an input conveys more information as it can potentially cause a more drastic change to the distance function. However, such an argument does not hold in the current study as we do not consider the learning of the distance function. Instead, we assume the input space is suitable for performing clustering, albeit for each cluster only a subset of the dimensions are relevant.

By summing up terms in the form of 1 - (normalized dispersion) rather than the dispersions themselves, an irrelevant dimension has relatively little effect to the objective score, thus algorithms can tolerate more errors. Also, within-cluster dispersion is measured by the distance from the cluster median rather than mean, which makes the function less affected by outliers.

4 The SSPC Algorithm

It is easy to observe that given a set of clusters $\{C_i\}_{i=1}^k$, the objective function ϕ is maximized when all dimensions with positive ϕ_{ij} are selected and all other dimensions are not selected. This leads to the deterministic dimension selection procedure in Figure 1.

The importance of SelectDim is that if we have an initial guess of the centers and relevant dimensions of the clusters, k-means(MacQueen, 1967) like iterative algorithms can be easily modified to identify projected clusters. Based on this

Figure 2 The outline of the SSPC algorithm.

Algorithm SSPC
1 Initialization: determine the seeds and relevant dimensions of each cluster
2 For each cluster, draw a medoid from the seeds
3 Assign every object in the dataset to the cluster (or outlier list) that gives the greatest improvement to the objective score
4 Call SelectDim(C_i) for each cluster C_i, and calculate the overall objective score
5 Record the clusters if they give the best objective score so far, restore the best clusters otherwise
6 Replace the cluster representatives (medoids or medians) of each cluster, then remove its members
7 Repeat 3-6 until no score improvements are observed for a certain number of iterations
End

idea, we present our new clustering algorithm SSPC (Semi-Supervised Projected Clustering) (Figure 2). We first give an overview of the algorithm, and then describe some components in detail in the coming subsections. It is a partitional algorithm similar to the k-medoids algorithms(Ng and Han, 1994). As in k-medoids, it has two main stages: initialization and iterative refinement. Each iteration is subdivided into two phases: object assignment and re-estimation.

Initialization: SSPC determines some seeds (potential medoids) and estimates the relevant dimensions of the corresponding clusters, and each cluster draws a medoid from them.

Object assignment: Every object in the dataset is assigned to the cluster that gives the greatest improvement to the objective score, where the value of $\tilde{\mu}_{ij}$ in Equation 6 is initially estimated by the projection of the medoid on v_j. If an object does not improve the ϕ_i score of any cluster, it will be put on the outlier list. Since the clusters being formed depend on the assignment order, when some ω's are non-zero, we randomly reorder all the objects before performing object assignment.

Re-estimation: After assigning all objects, the selected dimensions of each cluster are re-determined and the overall objective score is computed using the actual medians. If the new score is the best one encountered so far, the clusters will be recorded. Otherwise, the best clusters obtained so far will be restored. A bad cluster is then identified from the current best set of clusters, and a new medoid is selected for it with an attempt to improve the objective score in the next iteration. The medoid of each other cluster is replaced by the cluster median. For simplicity, we will call the medoid or median that is currently used to represent a cluster its "cluster representative". After replacing the old cluster representatives, the members of each cluster are removed, and a new iteration will start. The process repeats until the best objective score remains unchanged for a certain number of iterations.

There are several main differences between SSPC and the previous partitional approaches for projected clustering. First, the seeds are determined based on some domain knowledge if supplied, which is potentially more accurate. Second, the seeds of each cluster are associated with an estimated set of relevant dimensions

determined during initialization. When a seed is picked as the medoid of a cluster, the associated dimensions become the selected dimensions of the cluster. As to be seen later, this process does not rely on distance calculations that involve all dimensions or any user parameters whose values are hard to determine. This allows SSPC to identify low-dimensional clusters more accurately. Third, after each iteration, besides replacing a bad cluster representative by a new medoid, other cluster representatives are also replaced by the cluster medians to avoid problems due to the potential biased projected values of the medoids.

In the coming subsections we describe several important components of the algorithm: 1) Determining the values of the selection thresholds $\hat{s}_{ij}^2$; 2) The initialization process; 3) The replacement scheme for "bad" cluster representatives. After that, we analyze the algorithm in terms of its time complexity and the relationship between clustering accuracy and the amount of input knowledge.

4.1 Determining $\hat{s}_{ij}^2$

In order to use the SelectDim procedure and calculate the objective score ϕ, we need to determine the values of the selection thresholds $\hat{s}_{ij}^2$. Recall that a dimension v_j is selected for a cluster C_i if and only if its average squared distance from median is larger than $\hat{s}_{ij}^2$ when no penalty is applied. Since the expected variance of a random sample of a population is the variance of the population, if the within-cluster variance s_{ij}^2 is no smaller than the variance of the background distribution σ_j^2, the members of C_i are no more similar to each other along v_j than a group of random objects. Therefore, we use σ_j^2 (estimated by the sample variance s_j^2) as an upper bound of $\hat{s}_{ij}^2$.

We propose two schemes to set the actual value of $\hat{s}_{ij}^2$. The first scheme is to set it to ms_j^2, where $m \in (0, 1]$ is a user parameter. A smaller m tightens the selection criterion.

The second scheme is based on a probabilistic reasoning. Users need to specify a value p that bounds the maximum probability that a dimension irrelevant to a cluster is selected by chance. Suppose C_i is a cluster to which dimension v_j is irrelevant, then $p = Pr(s_{ij}^2 < \hat{s}_{ij}^2)$. If the sampling distribution of s_{ij}^2 has a known probability density function (PDF), the value of $\hat{s}_{ij}^2$ can be computed accordingly.

For example, suppose the background distributions are Gaussian. Then the random variable $\frac{(n_i-1)s_{ij}^2}{\sigma_j^2}$ has a chi-square distribution with $n_i - 1$ degrees of freedom. With p specified and σ_j^2 approximated by s_j^2, the value of $\hat{s}_{ij}^2$ can be computed from the inverse of the cumulative chi-square distribution.

The first scheme is more generic as it does not need to assume the properties of the background distributions. But in case the sampling distribution of s_{ij}^2 has a known PDF, the second scheme is more recommended as parameter p has a stronger intuitive meaning than m.

Although $\hat{s}_{ij}^2$ can take different values for different C_i (when parameter p is used) and v_j (in both cases), the objective function and the SelectDim procedure involves only one single user parameter (m or p). According to the experimental results to be presented in Section 5, the ranges of values of m and p that give good clustering results are usually very wide, which suggests that SSPC is quite robust. It is also possible to use any reasonable value (e.g. $0.3 \leq m \leq 0.7$, $0.01 \leq p \leq 0.2$)

in an initial test run, and then tune it down if the clusters appear to have too many selected dimensions, or tune it up if too many objects are discarded as outliers.

4.2 Initialization

During initialization, SSPC creates a number of seed groups, each containing a set of seeds that are expected to come from the same real cluster, and an estimated set of relevant dimensions of the real cluster. There are two kinds of seed groups: private and public. For a cluster with direct input knowledge (I^o and/or I^v), it is easier to form an accurate seed group in terms of both the seeds and the estimated relevant dimensions. A private seed group is thus created, which is solely used by the cluster. All clusters with no input knowledge share some large number of public seed groups, so that medoids can be drawn from different seed group combinations. Whenever a cluster needs to draw a new medoid, it is randomly drawn from the private seed group if it exists, or otherwise from one of the public seed groups not currently used by other clusters.

The order of seed group creation is important. Having created the seed groups of some clusters accurately, it becomes easier to accurately create the remaining seed groups. This is because objects that are close to the seeds of the previous seed groups in the estimated relevant subspaces are likely members of those clusters, which need not be considered when determining the seeds of the new seed groups. We will show in Section 4.5 that it is easier to create seed groups accurately for clusters with more input knowledge, which suggests that clusters with more input knowledge should have their seed groups created earlier. SSPC creates seed groups in the following order: 1) clusters with inputs in both I^o and I^v, 2) clusters with inputs in I^o only, 3) clusters with inputs in I^v only, 4) clusters with no inputs in either I^o or I^v. Within each category, clusters with larger amount of inputs are initialized earlier. Once an object is chosen as a seed, it will not be used in the creation of other seed groups.

In the following we discuss the details of the seed group creation process for the four cases. In the first three cases, a private seed group is created, and we will use C_i to denote the target cluster, G_i the resulting seed group, and I_i^o and I_i^v the sets of labeled objects and labeled dimensions for C_i respectively. Since G_i contains both a set of objects (the seeds) and a set of estimated relevant dimensions, we will sometimes treat it as a cluster for the sake of discussion.

4.2.1 Clusters with both input labeled objects and labeled dimensions

In this case, the center of C_i is likely to be located near the median of I_i^o. Also, if the set of objects in I_i^o is viewed as a temporary cluster $C_{i'}$, the dimensions with larger $\phi_{i'j}$ values are more likely to be relevant to C_i.

This leads to a two-step process of seed group creation. We first identify the seeds to form G_i, then we set the relevant dimensions of G_i as the union of I_i^v and the dimensions selected by SelectDim(G_i).

We developed a mechanism for the first step based on a simple idea. If we form a grid (multi-dimensional histogram) of the whole dataset using a fixed number of dimensions, then if the dimensions are all relevant to C_i, some cells will be found to contain a high density of objects, which correspond to regions close to the center of

C_i. If some of the dimensions are irrelevant to C_i, the density will be much lower. The basic idea is thus to build multiple grids using different sets of dimensions, and then pick the cell, among all built grids, with the highest object density. The objects in the cell will be chosen as the seeds of G_i.

There is, however, a complication in that a set of dimensions could be simultaneously relevant to multiple clusters. In such a case the grid being built will contain multiple peaks. To identify the peak corresponding to the center of the current cluster C_i, we perform a local hill-climbing search starting from the cell that contains the median of I_i^o rather than looking for the absolute peak of the whole grid. The local search also has the ability to correct the estimate of the cluster center should the input objects bias towards one side of C_i.

The number of dimensions used to build each grid should not be too large as the number of cells increases exponentially with respect to the number of building dimensions, which makes each cell to have too few objects and creates a heavy computational overhead. Normally a three-dimensional grid serves the purpose quite well.

Dimensions with greater chance of being relevant to C_i should have higher probabilities of being involved in grid building. Therefore we define a candidate set as the union of I_i^v and the dimensions selected by SelectDim($C_{i'}$), where the probability for each dimension v_j in the set to be used in building a grid is proportional to $\phi_{i'j}$.

As $\phi_{i'j}$ involves the computation of $s_{i'j}^2$, I_i^o should contain at least two objects.

4.2.2 Clusters with input labeled objects only

The seed group creation process is almost the same as in the previous case, except that only dimensions from SelectDim($C_{i'}$) are used in grid building and only those from SelectDim(G_i) are set as the relevant dimensions of G_i.

4.2.3 Clusters with input labeled dimensions only

In this case, the temporary cluster $C_{i'}$ cannot be formed. Only dimensions in I_i^v are involved in grid building, and each of them has the same probability of being used. Without $C_{i'}$, there is no starting point for the local hill-climbing search, so the seeds of G_i are the objects in the absolute peak of the whole grid.

4.2.4 Clusters with neither input labeled objects nor labeled dimensions

In this case, we look for objects that have cannot-links with some seeds from each private seed group. Such objects are good candidates for the members of the remaining clusters. For each such object, we look for objects that are must-linked to them directly or transitively. The resulting groups will be queued up in descending order of their sizes, and be used to initiate the creation of public seed groups as in the case of clusters with labeled objects as inputs only.

After this step, if we still need more public seed groups, we use a best-effort approach based on a simple idea: for those clusters with no seed groups created so far, their member objects are likely to be far away from all created seed groups. We thus use a max-min mechanism(Aggarwal et al., 1999) to identify an object

whose minimum distance to all seed group medians is maximum, where the distance calculations involve only the relevant dimensions of each seed group, normalized by the number of such dimensions.

The object is used as the starting point of the hill-climbing search. The probability for a dimension to be involved in grid building is proportional to the object density around the object in an one-dimensional histogram of the dimension.

In the extreme case that all clusters have no input knowledge, the initiating object of the first group is randomly drawn from the dataset.

4.3 Cluster Representatives Replacement

One of the big challenges of k-medoids algorithms is to avoid having the medoids of two clusters coming from the same real cluster. When that happens, the two clusters will compete for the ϕ_i score of the real cluster. On the other hand, one of the real clusters will not be represented by any cluster, and most of its members will be put on the outlier list.

One way to detect such a situation is to check if there is a cluster with a very low ϕ_i score as compared to the maximum achievable score and the scores of other clusters. Another way is to look for clusters that are very similar. In both cases, a bad cluster will be pinpointed and its cluster representative will be replaced by a new seed.

For each other cluster, although the medoid may really be a member of the real cluster, it may not be close to the cluster center along some relevant dimensions. As a result some real members located at the other side of the cluster may not be attracted to this cluster. SSPC attempts to improve the cluster by replacing the medoid with the median of the cluster, which is probably closer to the real cluster center.

4.4 Complexity

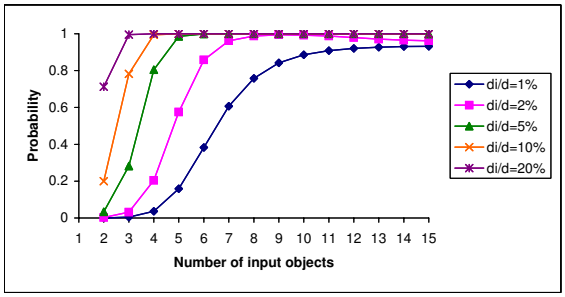
It can be shown(Yip et al., 2004) that SSPC has a time and space complexity of $O(knd)$ and $O(nd)$ respectively. The linear complexities make it more practical to cluster large datasets as compared to some other projected clustering algorithms such as DOC, HARP and ORCLUS.

4.5 How many inputs are needed?

In many real situations the amount of available domain knowledge is very limited, or it is very costly to obtain such knowledge. It is important to predict the relationship between the amount of input knowledge and the resulting clustering accuracy, so as to minimize the amount of input knowledge while getting a satisfactory accuracy.

We begin with the case where only labeled objects are available. Suppose a certain cluster C_i receives $|I_i^o|$ labeled objects. The objects form a temporary cluster, which is used to determine the grid-building dimensions. We want to estimate the probability that at least one grid is built from dimensions that are

Figure 3 The probability that at least one grid is formed by relevant dimensions only, when only labeled objects are available.



really relevant to C_i only, which is crucial to the accuracy of the seed group and the clustering accuracy in turn.

Assume parameter p is used to compute $\hat{s}^2_{ij}$. If dimension $v_{j'}$ is relevant to cluster C_i, then the probability that it is selected for C_i is $q = Pr(s^2_{ij'} < \hat{s}^2_{ij'}) = Pr(\frac{(|I^o_i|-1)s^2_{ij'}}{\sigma^2_{j'}} < \frac{(|I^o_i|-1)\hat{s}^2_{ij'}}{\sigma^2_{j'}})$, where $\sigma^2_{j'}$ is the variance of the underlying class along $v_{j'}$. If $\sigma^2_{j'}$ is estimated from the labeled objects and a histogram on $v_{j'}$, q can then be estimated from the cumulative chi-square distribution.

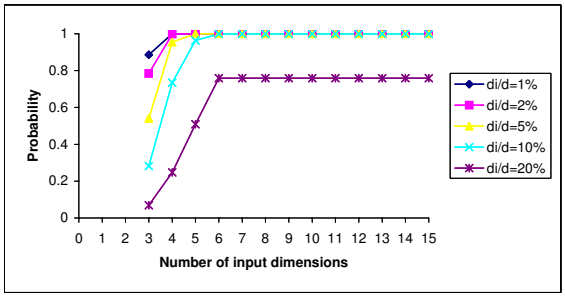
Let d_i be the number of relevant dimensions of C_i, t and f be the number of true and false positives (selected dimensions that are actually relevant/irrelevant to C_i) respectively. t is a binomial random variable with d_i trials and probability q, while f is a binomial random variable with $d - d_i$ trials and probability p. Suppose each dimension selected by the temporary cluster has the same chance of being used in grid building. The probability that a grid is built from c dimensions all being relevant to C_i is:

$$\alpha = \sum_{t=c}^{d_i} \sum_{f=0}^{d-d_i} \binom{d_i}{t} q^t (1-q)^{d_i - t} \binom{d-d_i}{f} p^f (1-p)^{d-d_i-f} \frac{\binom{t}{c}}{\binom{t+f}{c}}, \tag{6}$$

where $\binom{n}{r}$ represents the number of ways to pick r unordered outcomes from n possibilities. If g grids are built for each cluster, the probability that at least one of them involves only relevant dimensions is $1 - (1-\alpha)^g$. To visualize the change of this value with different input sizes, let us consider some real values to be used in the experiments in Section 5. Suppose $d = 3000$, $p = 0.01$, $c = 3$, $g = 20$, and the local-to-background variance ratio is 0.15. We vary $|I^o_i|$ from 2 to 15, and the ratio $\frac{d_i}{d}$, from 1% to 20%. Figure 3 shows the estimated probabilities that at least one grid is built from relevant dimensions only.

The figure shows that for a fixed $\frac{d_i}{d}$ ratio, having more input objects increases the probability of building a grid from relevant dimensions only. In addition, each curve is observed to have a sharp increase followed by a flattened region. This means it is possible to estimate the smallest amount of input that can lead to a near maximal accuracy. It is an exciting result to see that when $\frac{d_i}{d} = 5\%$, only 5 input items are enough to have an almost 100% guarantee that a grid will be built from relevant dimensions only. The figure also shows that for a fixed amount of inputs, the probability increases as $\frac{d_i}{d}$ increases, which suggests that input objects work better when the clusters have more relevant dimensions.

Figure 4 The probability that at least one grid with all c building dimensions being relevant to C_i only is formed, when only labeled dimensions are available.



Next, we consider the case where only labeled dimensions are available. Suppose dimension v_j is specified as relevant to a cluster C_i. If an one-dimensional histogram is built from v_j, we expect to find a peak at the center of C_i. If the cell with the highest object density is not close to the center of C_i, most probably v_j is also relevant to another cluster. Suppose there are k clusters, and on average each cluster has d_i relevant dimensions. Then the probability that v_j is also relevant to one or more other clusters is $1-(1-\frac{d_i}{d})^{k-1}$. Denote this probability as γ, the probability of forming a c-dimensional grid with all c dimensions being relevant to C_i only is $(1-\gamma)^c$. We want to estimate the probability of forming at least one such grid when g grids are built, which is a good indicator of the chance of forming an accurate seed group.

Suppose there are $|I_i^v|$ input dimensions, and no two grids are allowed to use exactly the same set of building dimensions. When $\binom{|I_i^v|}{c} \leq g$, the required probability is $1-(1-(1-\gamma)^c)^{\binom{|I_i^v|}{c}}$. When $\binom{|I_i^v|}{c} > g$, it becomes $1-(1-(1-\gamma)^c)^g$. Using the same parameter values as before, and setting $k=5$, the estimated probabilities at various $|I_i^o|$ and $\frac{d_i}{d}$ values are shown in Figure 4.

In general, the more labeled dimensions being supplied, the higher is the chance of forming a grid with all building dimensions being relevant to C_i only. The figure also reveals an interesting phenomenon: while labeled objects work better when $\frac{d_i}{d}$ is large, labeled dimensions work better when it is small as the chance for a single dimension to be relevant to multiple clusters is small. This suggests that when trying to identify clusters with extremely low dimensionality, which is the main focus of this study, it is more effective to use labeled dimensions as input knowledge.

Both results show that a very small amount of input knowledge could enhance the accuracy a lot. Since the two kinds of inputs complement each other, there is a synergy when they are supplied at the same time, provided the amount of input objects is not so small that causes a large amount of irrelevant dimensions to be used in building the grids. Some empirical results will be presented in the next section.

5 Experiments

5.1 Synthetic datset

In this section we present various experimental results on SSPC and the comparing projected clustering algorithms HARP(Yip et al., 2004) and PROCLUS(Aggarwal et al., 1999), using the non-projected k-medoids algorithm CLARANS(Ng and Han, 1994) as reference.

As far as we know, there exist no benchmark datasets that contain projected clusters as low dimensional as the ones that are of interest in this study. We therefore generate our own synthetic datasets in ways that are similar to some previous studies(Aggarwal et al., 1999; Yip, 2003), but with the parameters adjusted to produce datasets with the desired properties. We repeated each experiment 10 times, and report only the result that gives the best algorithm-specific objective score, in which the set of clusters produced is probably being adopted in a real situation when the real clusters are unknown. Meanwhile, we apply SSPC on real microarray data to show how the input knowledge improves the clustering performance.

The performance metric used to evaluate clustering results is the Adjusted Rand Index (ARI)(Yeung and Ruzzo, 2001). ARI measures how similar are the partitions of objects according to the real clusters (U) and a clustering result (V). Denote a, b, c and d as the number of object pairs that are in the same cluster in both U and V, in the same cluster in U but not V, in the same cluster in V but not U, and in different clusters in both U and V respectively, ARI is defined as follows:

$$ARI(U,V) = \frac{2(ad-bc)}{(a+b)(b+d)+(a+c)(c+d)} \tag{7}$$

The more similar are the two partitions, the larger is the ARI value. When U and V are identical, $ARI = 1$. When V is only as good as a random partition, $ARI = 0$.

If input knowledge is involved in a run of SSPC, the labeled objects are removed from the resulting clusters before computing the ARI values in order to eliminate the direct performance gain due to the input objects.

5.1.1 Raw accuracy

In the first set of experiments we compared the raw accuracy of the algorithms without input knowledge. A series of synthetic datasets were generated with $n = 1000$, $d = 100$ and $k = 5$. The actual average dimensionality of the clusters, l_{real}, varies from 5 to 40, accounting for $5\% - 40\%$ of the dataset dimensionality. The datasets were generated according to the data model described in Section 3, with the background distributions being uniform and the local distributions having variances ranging from $1\% - 10\%$ of the value range of the background distributions.

We set k to 5 for all algorithms, used default parameter values for HARP and CLARANS, and tried different values of the critical parameters of PROCLUS and SSPC. For PROCLUS, we tried 9 different values of l for each dataset (from 10% to 90% of the dataset dimensionality). For SSPC, 5 different values of m and p

Figure 5 The best raw accuracies (based on different parameter values) of the algorithms on datasets with various average cluster dimensionality.

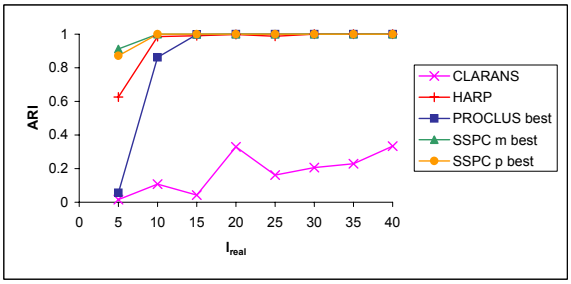
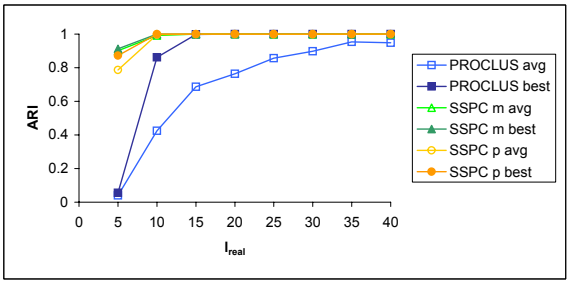


Figure 6 A comparison of the best and average raw accuracies (based on different parameter values) of PROCLUS and SSPC on datasets with various average cluster dimensionality.

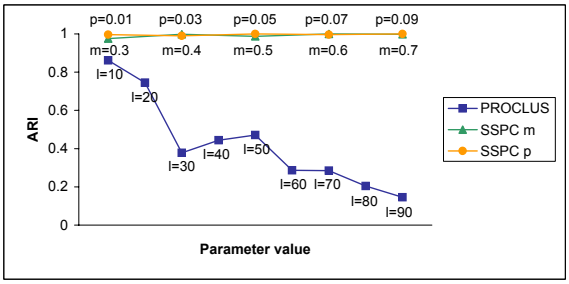


were used for each dataset ($0.3 \leq m \leq 0.7, 0.01 \leq p \leq 0.09$). The best results (the results with the highest ARI values) after trying different parameter values are shown in Figure 5. In Figure 6, the best and average results of PROCLUS and SSPC are shown for the comparison of their relative robustness.

The figure shows that all projected clustering algorithms performed well as compared to CLARANS when the cluster dimensionality is high. When the dataset dimensionality is as low as 5% of d, the performance of all three projected clustering algorithms went down, but SSPC has the mildest performance drop. It is somewhat unexpected that the raw performance of SSPC when parameter p is used is close to the performance when parameter m is used, given the background distributions are actually non-Gaussian. This may due to the fact that except the dimension selection procedure, SSPC makes no assumptions on the background distribution. The performance of the other parts of the algorithm may compensate for the invalid assumptions being made when parameter p is used.

Figure 6 shows that SSPC is more robust than PROCLUS in that the average accuracies over the use of different parameter values are much closer to the best accuracies. The individual clustering results when $l_{real} = 10$ are shown in Figure 7. PROCLUS performed well when the value of l was supplied correctly, but the performance went down as the input moved away from the true value. In contrast, SSPC performed well with the various parameter values being tried.

Figure 7 The accuracy of PROCLUS and SSPC on the dataset with $l_{real} = 10$ using various parameter values.



5.1.2 Performance with input knowledge

In this set of experiments, we further lower the average cluster dimensionality and see if the accuracy of SSPC can be improved by input knowledge.

We generated a dataset with $n = 150$, $d = 3000$, $k = 5$, and $l_{real} = 30$, i.e., 1% of d. The configuration highly resembles a gene expression dataset when the goal is to cluster the samples, and the number of relevant genes of each sample class is as low as 1% of d. We set $m = 0.5$, and tried 5 coverage ratios (fraction of clusters receiving input), and 8 input sizes. We first tried 4 input categories: no inputs, I^o only, I^v only, and both. For example, when coverage=0.6, only I^o and I^v are non-empty, and input size=4, $0.6 \times 5 = 3$ clusters receive input knowledge, each with 4 labeled objects and 4 labeled dimensions. No input is supplied for the other 2 clusters.

The input items were drawn randomly from the real cluster members and relevant dimensions. Each point in the coming figures is the median of 10 repeated runs with 10 independent sets of input.

Figure 8 shows the accuracy of SSPC when coverage=1. For reference, the ARI values of HARP and PROCLUS (with correct l value supplied) are 0.17 and 0.08 respectively, which are much lower than the raw accuracy of SSPC (at input size 0). In general, SSPC has a larger accuracy improvement when more input items are supplied. The accuracy becomes stable with 5 objects and 3 dimensions (which is equal to the default value of c, the number of building dimensions per grid). All these observations are consistent with the analysis in Section 4.5. The accuracy of SSPC appears to be more stable with labeled dimensions as inputs. In particular, an accuracy lower than the raw accuracy is observed when only 2 labeled objects are supplied to each cluster. This is due to the large probability that the two objects are close to each other along many irrelevant dimensions, which misleads the dimension selection procedure (see Figure 3). In contrast, the probability for a pair of dimensions to be relevant to multiple clusters is much lower due to the low average cluster dimensionality, which results in an observable accuracy improvement when 2 labeled dimensions are supplied.

Figure 9 shows the accuracy of SSPC with various coverage values, when the input sizes are 3 and 6 respectively. Again, there is a general trend of increasing accuracy when the coverage increases, and the increasing trend is more stable at the larger input size of 6. An interesting observation from Figure 9b is that the peak performance is reached at 60% coverage, which suggests that it is not necessary to

Figure 8 The accuracy of SSPC with various amount of input knowledge when the coverage is 1.

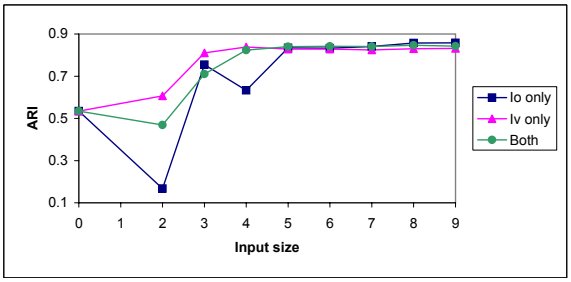
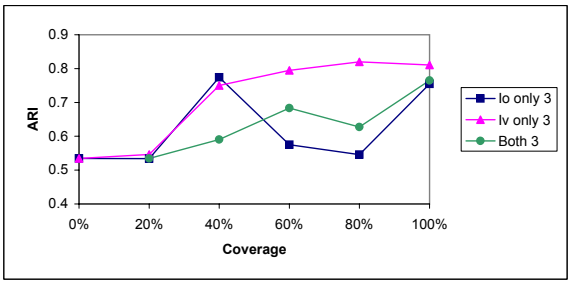
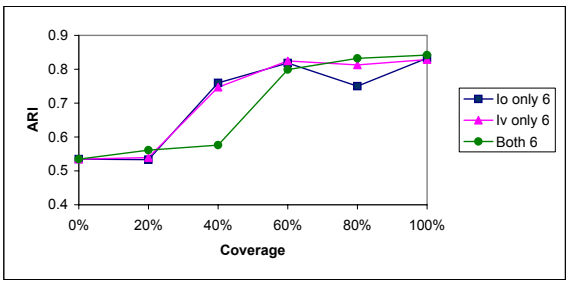


Figure 9 The accuracy of SSPC with various coverage of input knowledge.



(a) When input size is 3.



(b) When input size is 6.

input domain knowledge to every cluster. By using the max-min mechanism (Section 4.2.4), clusters with no input knowledge could also locate their cluster centers provided the seed groups of the other clusters are created accurately. During object assignment, if members of the clusters with input knowledge are assigned correctly, the chance for the remaining clusters to identify their members and relevant dimensions correctly is also increased.

We then studied the effectiveness of must-links and cannot-links. We generated a new dataset with the same number of objects and dimensions, but much wider variances of the Gaussians. In other words, this dataset is more difficult to cluster than the one we just used to test the effectiveness of labeled objects and labeled dimensions. The purpose of using a more difficult dataset is to illustrate the additional benefits brought by the link inputs.

For each of the three original input combinations (I^o only, I^v only, I^o and I^v), we compared the performance of SSPC with no links as inputs, with M^o only, with C^o only, and with both. Therefore in total 12 input type combinations were tried. We studied the trend of the accuracy of SSPC with different input sizes. At a particular input size s, if M^o was used, s objects were selected from each cluster and all pairs of objects from the same cluster became the must-links. If C^o was used, s objects were selected from each cluster and all pairs of objects from different clusters became the cannot-links.

Figure 10 shows the best accuracies across multiple runs of SSPC. The figure shows that the accuracy of SSPC was improved by the link inputs in two ways. First, for most of the input sizes, having links as inputs gave a better clustering accuracy than when no links were supplied. Second, the link inputs allowed the peak accuracy of SSPC to be raised, which can be seen from the higher ARI scores at large input sizes (7 or larger).

5.1.3 Data with multiple possible groupings

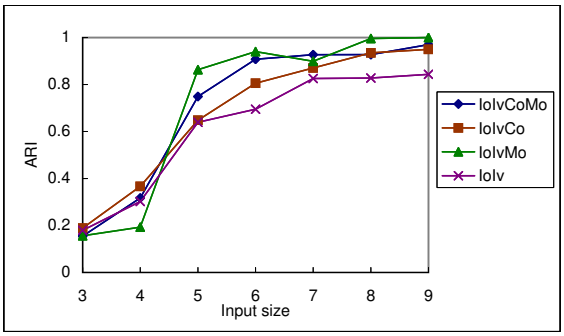
As discussed in Section 1, an important application of semi-supervised clustering is to produce different desired clusters based on different input knowledge. In this set of experiments we verify this capability of SSPC. We generated two datasets, each with $n = 150$, $d = 1500$, $k = 5$ and $l_{real} = 30$. The members and relevant dimensions of the clusters in the two datasets are independent. We then combined the two datasets to produce a dataset with 3000 dimensions, where the first 1500 come from the first original dataset and the last 1500 come from the second. The average cluster dimensionality thus remains at 1% of d. We then tested the accuracy of HARP, PROCLUS and SSPC on the dataset, with correct l value supplied to PROCLUS. For SSPC, we tested its accuracy in three different scenarios: without input (raw accuracy), input based on the knowledge of the first original dataset, and input based on the knowledge of the second original dataset. The ARI values of the algorithms computed from the actual clusters of the two original datasets are shown in Figure 11.

The performance of HARP is seriously affected by the simultaneous existence of two possible groupings. Objects not in the same cluster can be close to each other along many dimensions (as they do belong to the same cluster in the other grouping), which ruins the threshold loosening mechanism of HARP. The performance of PROCLUS is better, but is still not very encouraging. The raw accuracy of SSPC is better than HARP and PROCLUS when evaluated by the first set of clusters, but worse when evaluated by the second set. This shows that without any external input, SSPC tends to form clusters that are more similar to the first set. But as some external inputs were supplied, the accuracy of SSPC was significantly improved in both cases. The results confirm the importance of external inputs in guiding the formation of some desired clusters when there are multiple meaningful groupings.

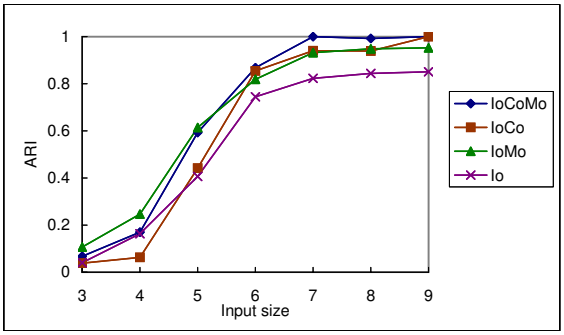
5.1.4 Scalability

Figures 12a and 12b show the execution time of 10 repeated runs of SSPC with an increasing dataset size (n) and dimensionality (d) respectively (without

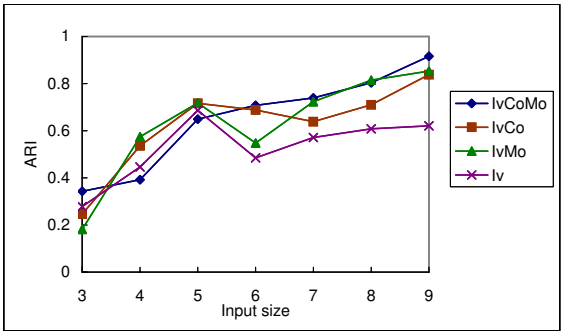
Figure 10 The accuracy of SSPC with and without link inputs.



(a) With I^o and I^v.



(b) With I^o.



(c) With I^v.

Figure 11 The accuracy of the algorithms on the dataset with two possible groupings.

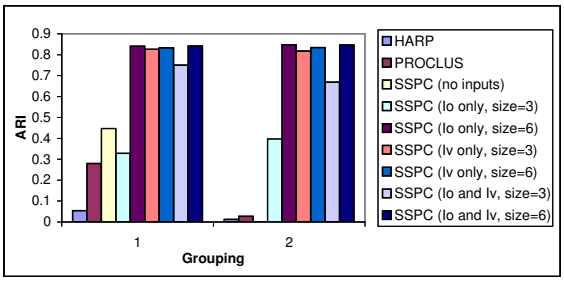
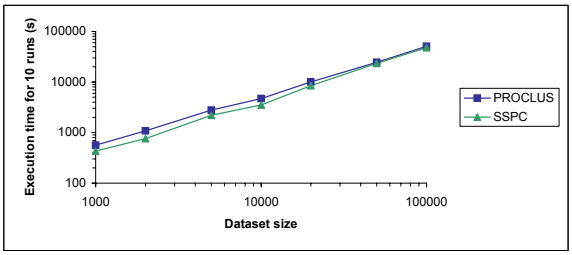
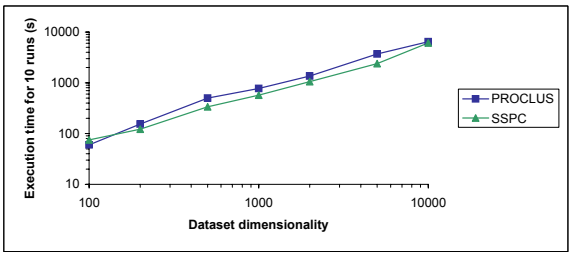


Figure 12 The execution time of 10 repeated runs of PROCLUS and SSPC.

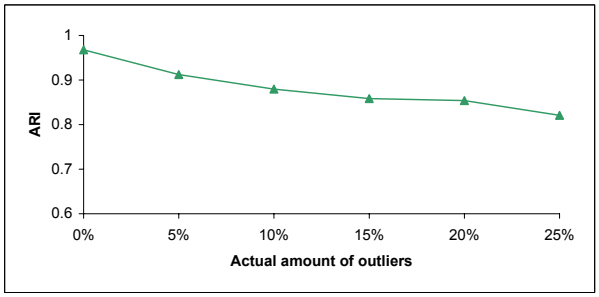


(a) With an increasing n.



(b) With an increasing d.

Figure 13 The accuracy of SSPC on datasets with various amount of outliers.



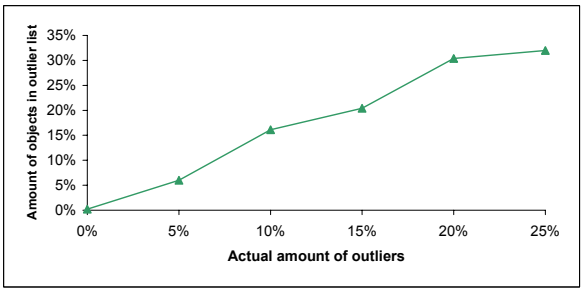
knowledge input), using the execution time of PROCLUS as reference. The figures confirm the linear time complexity of SSPC with respect to both n and d. Its speed is comparable to PROCLUS in our implementations.

5.1.5 Outlier Immunity

Finally we studied how SSPC is affected by outliers. A series of synthetic datasets were generated with $n = 1000$, $d = 100$, $k = 5$, and $l_{real} = 5$. The amount of outliers varies from 0% to 25%. We set m to 0.5. The clustering accuracies and the amount of objects on the outlier lists (without knowledge input) are shown in Figure 13 and Figure 14 respectively.

The results show that SSPC has a high noise-immunity, with only moderate accuracy decrease as the amount of outliers increases. The amount of objects detected as outliers also highly resembles the actual amount of outliers in the datasets.

Figure 14 The amount of objects put in the outlier list by SSPC when clustering the datasets with various amount of outliers.



5.2 Real dataset

In this subsection, we apply SSPC to a microarray data set SRBCT which is a gene expression data set presented by Khan et al. (Khan et al., 2001). It contains the expression levels of 2308 genes for 63 Small Round Blue Cells Tumor patients belonging to one of the 4 categories: Ewing family of tumors (EWS), rhabdomyosarcoma (RMS), neuroblastoma (NB) and non-Hodgkin lymphoma (BL). For this data set, we clustered the samples with the genes as the input dimensions. ARI is also used here to evaluate the clustering results by calculating the correspondence between the clusters generated by SSPC and the inherent categories of the data set.

Various of experiments on SRBCT were conducted to show the performance of SSPC with input knowledge. For the relevant objects (samples), we randomly draw them from the four categories. Meanwhile, some genes are identified for each categories (see Khan et al. (2001)) and the input relevant dimensions were drawn from these genes. In the experiments, the parameter m was set to 0.5. Each point in the coming figures is the average of 10 repeated runs with 10 independent sets of input.

Figure 15 shows the performance of SSPC on SRBCT with different input sizes when coverage=1. When there is no input knowledge (i.e., input size is 0), SSPC obtained a bit worse clustering results. However, the clustering performance was significantly improved when the input items are supplied. From Figure 15, we can see that the clustering accuracy (measured by ARI) becomes stable when the number of input labeled dimensions is 6. Meanwhile, the clustering accuracy (measured by ARI) becomes better with more input labeled objects.

Figure 16 shows the clustering accuracy of SSPC with various coverage values, when the input sizes are 3 and 7 respectively. In general, there is a general trend of increase of clustering accuracy when the coverage increases. Again, an interesting observation from Figure 16b is that the peak performance is reached at 50% coverage (in this case, only 2 (50% $\times$ 4) clusters are covered), which suggests that it is not necessary to input domain knowledge to every cluster. This result is similar to the experimental results of synthetic data sets, for that the chance of identifying the clusters with no input knowledge is increased when the members of the clusters with input knowledge could be assigned correctly.

Also, we studied the effectiveness of must-links and cannot-links on clustering microarray data. Again, there are 12 input type combinations for M^o and C^o based

Figure 15 The accuracy of SSPC on SRBCT with various amount of input knowledge when the coverage is 1.

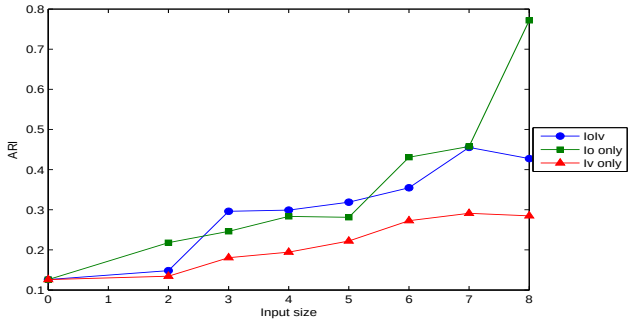
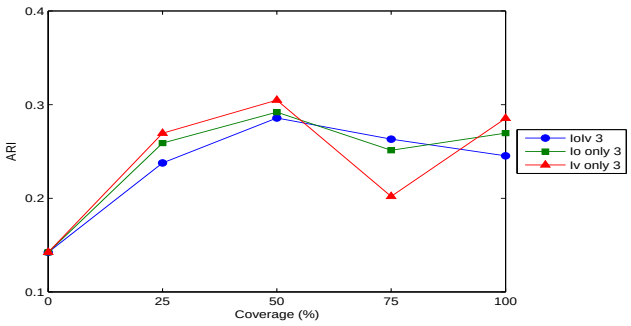
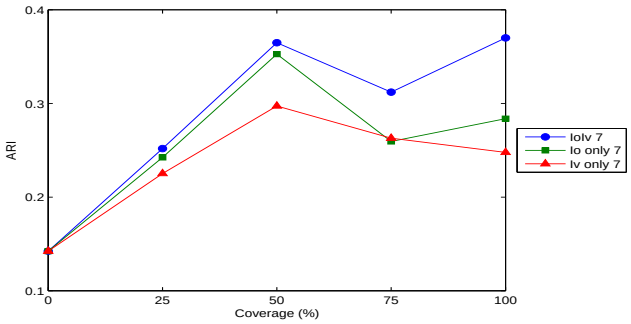


Figure 16 The accuracy of SSPC on SRBCT with various coverage of input knowledge.

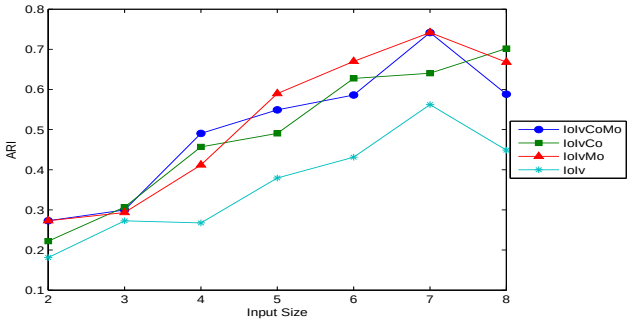


(a) When input size is 3.

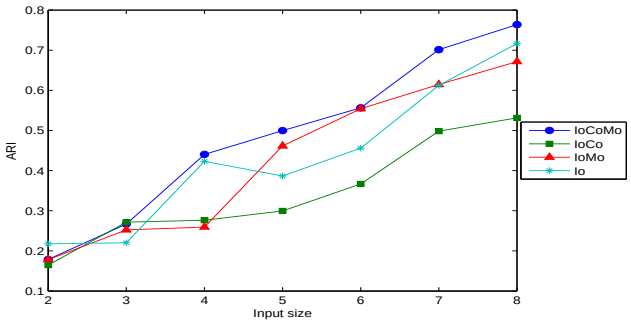


(b) When input size is 7.

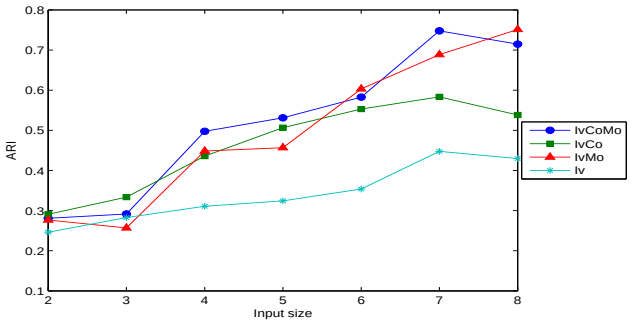
Figure 17 The Accuracy of SSPC on SRBCT with and without link inputs.



(a) With I^o and I^v.



(b) With I^o.



(c) With I^v.

on the combinations of I^o and I^v. Figure 17 shows that the performance of SSPC was improved by the link inputs in two ways. For most of the input sizes, having links as inputs gave a better clustering accuracy than when no links were supplied. In particular, the clustering accuracy reaches its peak point that the number of input size is 7. This result also suggests that it is not necessary to input much more domain knowledge for much better clustering result, i.e., small knowledge can improve the clustering performance.

6 Summary and Future Extensions

In this paper, we have discussed some potential limitations of some existing projected clustering algorithms, including their inability to detect clusters with very low dimensionality, the use of user parameters whose proper values are hard to determine, and the potential accuracy drop when improper parameter values are supplied. We have proposed a new projected clustering algorithm that is robust and is able to detect clusters of extremely low dimensionality as it uses a robust objective function and avoids distance calculations that involve all the dimensions. In addition, we have proposed ways to utilize any available domain knowledge in the form of labeled objects and labeled dimensions. Experimental results show that there is a clear accuracy improvement when some input knowledge is incorporated in the clustering process. The peak performance is readily reached when only a small amount of knowledge is supplied, and when the knowledge covers only some of the classes.

There are some interesting directions for further study. The current study is focused on distance-based clustering, i.e., the similarity between different objects is measured by a distance function. There are more and more applications where similarity is measured by other means, such as the rise and fall pattern of values. Since most semi-supervised clustering algorithms are based on some distance-based methods such as k-means and k-medoids, some adaptations may be required when trying to incorporate domain knowledge in pattern-based clustering.

It is also interesting to study the case where one class corresponds to multiple clusters. In(Klein et al., 2002), an interesting algorithm is proposed that modifies the distance matrix such that objects of the same class move towards each other to form a single cluster. The more general approach that allows the formation of multiple clusters per class is not yet fully studied.

While the various kinds of inputs all result in accuracy improvements in the experiments, the relative effectiveness and the interactions between them are still unclear. Some more theoretical analysis could probably help gain insights for improving the algorithm.

Acknowledgements

The authors would like to thank the support from Hong Kong Research Grant Council Grant Number HKU 7117/05E and HKBU FRGs.

References

Pomeroy, S.L., Tamayo, P., Gaasenbeek, M., Sturla, L.M., Angelo, M., McLaughlin, M.E., Kim, J.Y.H., Goumnerovak, L.C., Black, P.M., Lau,C., Allen, J.C., Zagzag, D., Olson, J.M., Curran, T., Wetmore, C., Biegel, J.A., Poggio, T., Mukherjee, S., Rifkin, R., Califano, A., Stolovitzky, G., Louis, D.N., Mesirov, J.P., Lander, E.S., and Golub, T.R. (2002) 'Prediction of central nervous system embryonal tumour outcome based on gene expression', *Nature*, Vol. 415, pp.436–442.

Procopiuc, C.M., Jones, M., Agarwal, P.K. and Murali, T.M. (2002) 'A Monte Carlo algorithm for fast projective clustering', *ACM SIGMOD International Confer-*

ence on Management of Data.

Wang, H., Wang, W., Yang, J. and Yu, P.S. (2002) 'Clustering by pattern similarity in large data sets',*ACM SIGMOD International Conference on Management of Data.*

Yip, K.Y. (2003) 'HARP: A practical projected clustering algorithm for mining gene expression data', *Master's thesis*, Department of Computer Science, University of Hong Kong, $http://www.cs.hku.hk/\ ylyip/papers/HARP_HKUCS2003.pdf$.

Aggarwal, C.C., Procopiuc, C., Wolf, J.L., Yu, P.S. and Park, J.S. (1999) 'Fast algorithms for projected clustering',*ACM SIGMOD International Conference on Management of Data.*

Yip, K.Y., Cheung, D.W., and Ng, M.K. (2004) 'HARP: A practical projected clustering algorithm',*IEEE Transactions on Knowledge and Data Engineering (TKDE)*, Vol. 16, No. 11, pp. 1387–1397.

Demiriz, A., Bennett, K.P. and Embrechts, M.J. (1999) 'Semi-supervised clustering using genetic algorithms', *Artificial Neural Networks In Engineering.*

Wagstaff, K., Cardie, C., Rogers, S. and Schroedl, S. (2001) 'Constrained k-means clustering with background knowledge', *Proceedings of the Eighteenth International Conference on Machine Learning.*

Cohn, D., Caruana, R. and McCallum, A. (2003) 'Semi-supervised clustering with user feedback', *Cornell University, Tech. Rep. TR2003-1892.*

Wagstaff, K. and Cardie, C. (2003) 'Clustering with instance-level constraints', *Proceedings of the Seventeenth International Conference on Machine Learning.*

Yip, K.Y., Cheung, D.W. and Ng, M.K. (2005) 'On discovery of extremely low-dimensional clusters using semi-supervised projected clustering', *21st International Conference on Data Engineering (ICDE'05)*, pp. 329–340.

Ng, R.T. and Han, J. (1994) 'Efficient and effective clustering methods for spatial data mining', *20th International Conference on Very Large Data Bases*, September 12–15, Santiago.

Aggarwal, C.C. and Yu, P.S. (2000) 'Finding generalized projected clusters in high dimensional spaces', *ACM SIGMOD International Conference on Management of Data.*

Aggarwal, C.C., Han, J., Wang, J. and Yu, P.S. (2003) 'A framework for projected clustering of high dimensional data streams', *Proceedings of the 2003 International Conference on Very Large Data Bases (VLDB'03).*

Agrawal, R., Gehrke, J., Gunopulos, D. and Raghavan, P. (1998) 'Automatic subspace clustering of high dimensional data for data mining applications', *ACM SIGMOD International Conference on Management of Data.*

Cheng, Y. and Church, G.M. (2000) 'Biclustering of expression data', *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology.*

Basu, S., Banerjee, A. and Mooney, R. (2002) 'Semi-supervised clustering by seeding', *Proceedings of the Nineteenth International Conference on Machine Learning.*

Basu, S., Banerjee, A. and Mooney, R. (2004) 'Active semi-supervision for pairwise constrained clustering', *Proceedings of the SIAM International Conference on Data Mining.*

Bilenko,M., Basu, S. and Mooney, R. (2004) 'Integrating constraints and metric learning in semi-supervised clustering', *Proceedings of the Twenty-First International Conference on Machine Learning.*

Klein, D., Kamvar, S.D. and Manning, C.D. (2002) ''From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering', *Proceedings of the Nineteenth International Conference on Machine Learning.*

Basu, S. Bilenko,M., and Mooney, R. (2004) 'A probabilistic framework for semi-supervised clustering', *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Talavera L. and Bejar, J. (1999) 'Integrating declarative knowledge in hierarchical clustering tasks', *International Symposium on Intelligent Data Analysis.*

Xing, E.P., Ng, A.Y., Jordan, M.I. and Russell, S. (2003) 'Distance metric learning with application to clustering with side-information', *Advances in Neural Information Processing Systems.*

Ratsaby, J. and Venkatesh, S.S. (1995) 'Learning from a mixture of labeled and unlabeled examples with parametric side information', *Eighth Annual Conference on Learning Theory (COLT).*

Blum, A. and Mitchell, T. (1998) 'Combining labeled and unlabeled data with co-training', *Eleventh Annual Conference on Learning Theory (COLT).*

Szummer, M. and Jaakkola, T. (2001) 'Partially labeled classification with Markov random walks', *2001 Neural Information Processing Systems (NIPS) Conference.*

Pei, J., Zhang, X., Cho, M., Wang, H. and Yu, P.S. (2003) 'MaPle: A fast algorithm for maximal pattern-based clustering', *IEEE International Conference on Data Mining.*

Quinlan, J.R. (1993) 'C4.5 Programs for Machine Learning', *Morgan Kaufmann.*

Basu, S. Bilenko,M., and Mooney, R. (2003) 'Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering', *ICML Workshop on the Continum from Labeled to Unlabeled Data in Machine Learning and Data Mining.*

MacQueen, J.B. (1963) 'Some methods for classification and analysis of multivariate observations', *5th Berkeley Symposium on Mathematical Statistics and Probability.*

Yip, K.Y., Cheung, D.W. and Ng, M.K. (2004) 'On discovery of extremely low-dimensional clusters using semi-supervised projected clustering', *HKU CS, Tech. Rep. TR-2004-08*, *http* : *//www.cs.hku.hk/research/techreps/document/TR* − 2004 − 08.*pdf*.

Yeung, K. and Ruzzo, W. (2001) 'An empirical study on principal component analysis for clustering gene expression data', *Bioinformatics*, Vol. 17, No. 9, pp.763–774.

J. Khan, J., Wei, J., Ringner, M., Saal, L., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C., Peterson, C. and Meltzer, P. (2001) 'Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks', *Nature medicine*, Vol. 7, No. 6, pp.673–679.

Yeung, K. and Ruzzo, W. (2002) 'Machine learning methods applied to DNA microarray data can improve the diagnosis of Cancer', *ACM SIGKDD Explorations*, Vol. 5, No. 2, pp.48–55.