UNIVERSITY OF
LIVERPOOL

Data Clustering and Partial Supervision with Some Parallel Developments

A thesis submitted in accordance with the requirements of the

University of Liverpool for the degree of Doctor of Philosophy

By

Sameh A. Salem

December 2007

TEXT BOUND INTO

THE SPINE

to my beloved mother

Declaration

The work in this thesis is based on research carried out at the University of Liverpool. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Sameh A. Salem

Liverpool, United Kingdom.

Abstract

Data Clustering and Partial Supervision with Some Parallel Developments

by Sameh A. Salem

Clustering is an important and irreplaceable step towards the search for structures in the data. Many different clustering algorithms have been proposed. Yet, the sources of variability in most clustering algorithms affect the reliability of their results. Moreover, the majority tend to be based on the knowledge of the number of clusters as one of the input parameters. Unfortunately, there are many scenarios, where this knowledge may not be available. In addition, clustering algorithms are very computationally intensive which leads to a major challenging problem in scaling up to large datasets. This thesis gives possible solutions for such problems.

First, new measures - called clustering performance measures (*CPMs*) - for assessing the reliability of a clustering algorithm are introduced. These *CPMs* can be used to evaluate: 1) clustering algorithms that have a structure bias to certain type of data distribution as well as those that have no such biases, 2) clustering algorithms that have initialisation dependency as well as the clustering algorithms that have a unique solution for a given set of parameter values with no initialisation dependency.

Then, a novel clustering algorithm, which is a RAdius based Clustering ALgorithm (*RACAL*), is proposed. *RACAL* uses a distance based principle to map the distributions of the data assuming that clusters are determined by a distance parameter, without having to

specify the number of clusters. Furthermore, *RACAL* is enhanced by a validity index to choose the best clustering result, i.e. result has compact clusters with wide cluster separations, for a given input parameter. Comparisons with other clustering algorithms indicate the applicability and reliability of the proposed clustering algorithm. Additionally, an adaptive partial supervision strategy is proposed for using in conjunction with *RACAL* to make it act as a classifier. Results from *RACAL* with partial supervision, *RACAL-PS*, indicate its robustness in classification. Additionally, a parallel version of *RACAL* (*P-RACAL*) is proposed. The parallel evaluations of *P-RACAL* indicate that *P-RACAL* is scalable in terms of speedup and scaleup, which gives the ability to handle large datasets of high dimensions in a reasonable time.

Next, a novel clustering algorithm, which achieves clustering without any control of cluster sizes, is introduced. This algorithm, which is called Nearest Neighbour Clustering Algorithm (*NNCA*), uses the same concept as the K-Nearest Neighbour (*KNN*) classifier with the advantage that the algorithm needs no training set and it is completely unsupervised. Additionally, *NNCA* is augmented with a partial supervision strategy, *NNCA-PS*, to act as a classifier. Comparisons with other methods indicate the robustness of the proposed method in classification. Additionally, experiments on parallel environment indicate the suitability and scalability of the parallel *NNCA*, *P-NNCA*, in handling large datasets.

Further investigations on more challenging data are carried out. In this context, microarray data is considered. In such data, the number of clusters is not clearly defined. This points directly towards the clustering algorithms that does not require the knowledge of the number of clusters. Therefore, the efficacy of one of these algorithms is examined. Finally, a novel integrated clustering performance measure (*ICPM*) is proposed to be used as a guideline for choosing the proper clustering algorithm that has the ability to extract useful biological information in a particular dataset.

Acknowledgements

There are many people that I would like to thank. In particular, I gratefully acknowledge the constant support and guidance from my supervisor Prof. Asoke K. Nandi throughout this research. I would like to thank all my colleagues in the Signal Processing and Communication (SPC) Group. This group has been an enjoyable place and a source of friendships as well as good advice and collaboration. A special thanks to Dr. Lindsay Jack for his fruitful discussions, suggestions, and experience in system administration at the early stages of my research. Thanks must go to Dr. Sonu Punnoose, my system administrator companion, whose positive attitude, sense of humour and help were really appreciated. I also wish to express my thanks to Nancy, Alfonso, Jenny, TingTing, Liang, and Dr. Luciano Sarperi who always made sure that our network is working properly.

I am deeply indebted to the Egyptian Ministry of Higher Education for the financial support for this research. I am ever-thankful to my wife Nancy for her love, support, help, and encouragement. I am forever indebted to her. My regret and apology to my kids, Ahmed and Mohammed, for leaving them in Egypt during the last three years.

A special thanks to my father for his wishes, support and enthusiasm. Thanks should also go to my brothers Ayman and Khaled. I owe special thanks to my mother-in-law for her care and for making my life easier, it is truly appreciated. Due acknowledgement must also be made to my father-in-law for his continuous wishes, and how could I forget Tamer, Sally, and Fatty, very grateful thanks to all of them. I would like to thank my best friend Mohammed Nagy for his care of my mother during her last days.

Lastly, very late thanks to my mother who died in Egypt while I was away. I would like to dedicate this thesis to her.

Contents

List of Figures

List of Tables

Acronyms

AIC	Akaike's Information Criterion
BIC	Bayesian Inference Criterion
CAIC	Consistent Akaike's Information Criterion
CAD	Computer-Aided Diagnosis
CPMs	Clustering Performance Measures
CPM_1	Clustering Performance Measure 1
CPM_2	Clustering Performance Measure 2
CPU	Central Processing Unit
FLDA	Fisher Linear Discriminant Analysis
FOV	Field Of View
FPR	False Positive Rates
GP	Genetic Programming
HC	Hierarchical Clustering
ICPM	Integrated Clustering Performance Measure
KNN	K-Nearest Neighbour
MDC	Minimum Distance Classifier
MDL	Minimum Description Length
MLP	Multi-Layer Percepton

MPI	Message Passing Interface
NNCA	Nearest Neighbour Clustering Algorithm
NNCA-PS	Nearest Neighbour Clustering Algorithm with Partial Supervision
PAM	Partition Around Medoids
PCA	Principal Component Analysis
P-NNCA	Parallel Nearest Neighbour Clustering Algorithm
P-RACAL	Parallel RAdius-based Clustering ALgorithm
QAM	Quadrature Amplitude Modulation
RACAL	RAdius-based Clustering ALgorithm
RACAL-PS	RAdius-based Clustering ALgorithm with Partial Supervision
RAM	Random Access Memory
ROC	Receiver Operating Characteristic
SNR	Signal to Noise Ratio
SOM	Self Organising Maps
SOON	Self-Organising Oscillator Network
SR	Sampling Ratio
SVM	Support Vector Machine
TPR	True Positive Rates

Chapter 1

Introduction

1.1 Motivations

TODAY, the world is full of data, and the continuous advancement in computer technology makes creation of large amount of data faster and easier way. Clustering is one such way to summarise this huge amount of data into a small number of groups or categories. Clustering is a common technique for data analysis, which is used in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics.

Clustering algorithms offer several advantages over manual grouping processes. Yet, the sources of variability in most clustering algorithms affect the reliability of their results. The variability in clustering results is either produced by some stochastic components or setting of the input parameter(s). Therefore, there is a need for performance measures that are able to examine not only the quality of the clustering results but also the stability of results in a quantitive manner. This offers guidelines for choosing the proper clustering algorithm for a particular dataset, and the parameter settings which give the most promising results.

Many different clustering algorithms exist, and these can be used to find different numbers of clusters depending on the requirements of the particular problem. A standard requirement in existing clustering algorithms is to specify the number of clusters. In well

bounded and understood datasets, it is relatively easy to determine how many different clusters are required. Once this number is known, then it is relatively easy to perform the clustering and interpret the results. However, when the data is either noisy or not easily separable (which is often the case with many different real-world datasets), it can be much more difficult to determine with confidence the true number of clusters for the clustering algorithm. All these point towards new techniques for clustering to alleviate the requirement of the number of clusters.

Clustering is unsupervised classification where there are no predefined classes (labels) and no a priori knowledge of the data, while supervised classification requires a complete knowledge of the data where the class label and the number of classes (labels) are predefined. The process of acquiring a complete knowledge of the data is often limited and always an expensive and error-prone task that requires time and human intervention. Therefore, the enhancement of the clustering process with a small proportion of labelled data can guide the clustering process of the unlabelled data.

Clustering algorithms are computationally intensive, particularly when these algorithms are used to analyse large amounts of data. When a fast serial machine can not deliver results in a reasonable time, a possible approach to reduce the processing time is based on the implementation of clustering algorithms on scalable parallel computers. This can provide the appropriate setting to execute efficiently clustering algorithms for extracting knowledge from large-scale datasets.

1.2 Contributions

The following is a summary of the original contributions of my research in the field of data clustering:

- Novel measures - called clustering performance measures (*CPMs*) - for assessing the reliability of a clustering algorithm are introduced. These *CPMs* can be used to evaluate: 1) clustering algorithms that have a structure bias to a certain type of data distribution as well as those that have no such bias, 2) clustering algorithms that

have initialisation dependency as well as the clustering algorithms that have a unique solution for a given set of parameter values with no initialisation dependency.

- A novel clustering algorithm, called RAdius based Clustering ALgorithm (*RACAL*), is proposed. *RACAL* uses a distance based principle to map the distributions of the data assuming that clusters are determined by a distance parameter, without having to specify the number of clusters. Furthermore, *RACAL* is enhanced by a validity index to choose the best clustering results for a given input parameter. Additionally, an adaptive partial supervision strategy is proposed for use in conjunction with *RACAL* to make it act as a classifier.

- A novel clustering algorithm, which achieves clustering without any control of cluster sizes, is introduced. This algorithm, which is called Nearest Neighbour Clustering Algorithm (*NNCA*), uses the same concept as the K-Nearest Neighbour (*KNN*) classifier with the advantage that the algorithm needs no training set and it is completely unsupervised. Additionally, *NNCA* is augmented with a partial supervision strategy to act as a classifier.

- Parallel design and implementations of *RACAL* and *NNCA* clustering algorithms to handle large datasets of high dimensions and reduce the clustering time, which helps in improving the clustering quality as well as improving the flexibility in data exploration.

- Investigation of the use of a recently developed clustering algorithm, Self-Organising Oscillator Network (*SOON*) that has biological roots and does not require the knowledge of the number of clusters, in analysing microarray data.

- Development of a novel integrated clustering performance measure (*ICPM*) to be used as a guideline for choosing the proper clustering algorithm that has the ability to extract useful biological information in a particular microarray dataset.

Figure 1.1 shows a graph that describes the novel contributions of this thesis in the field of data clustering.

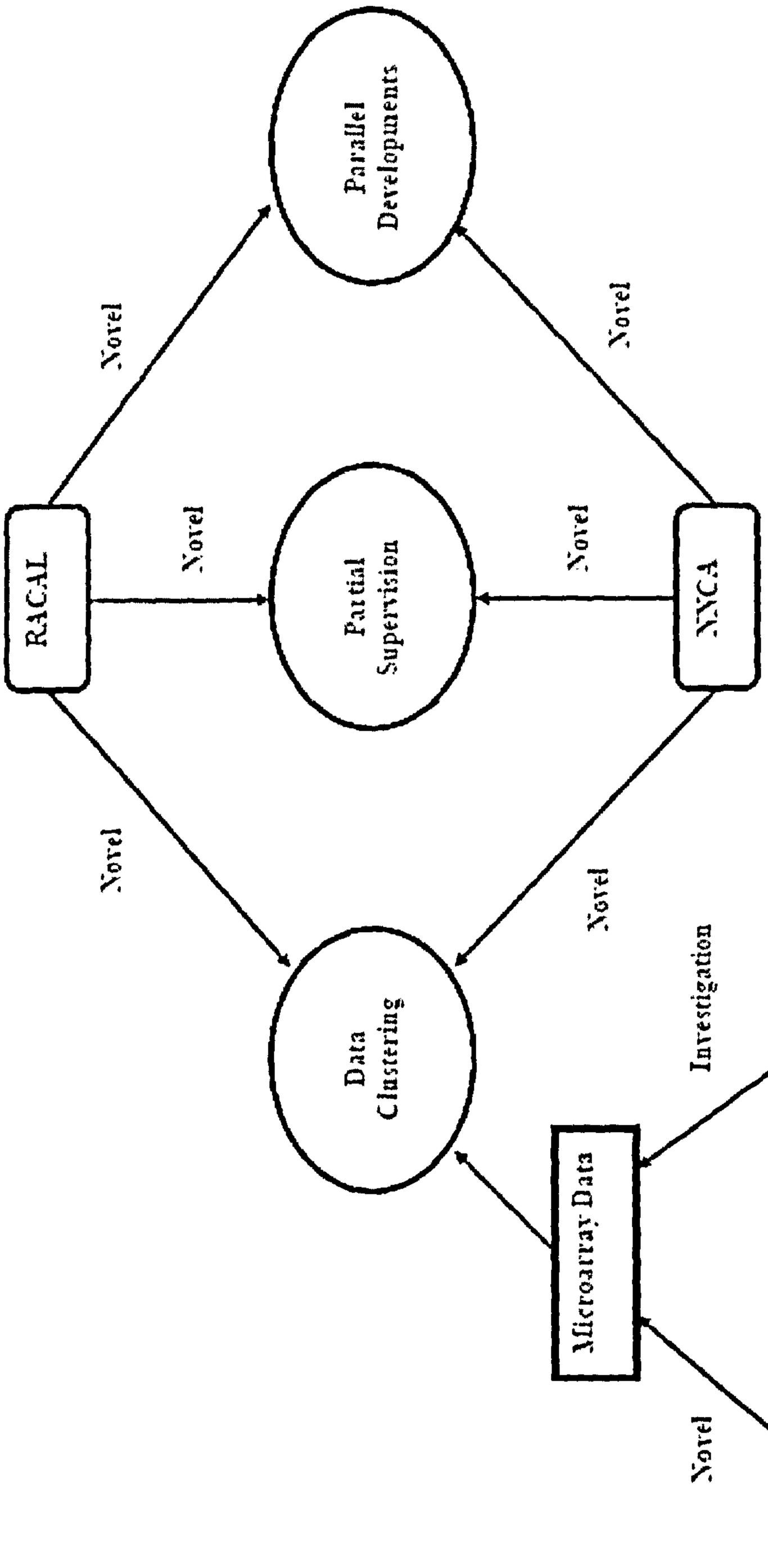


Figure 1.1: A descriptive graph for the thesis contributions.

1.3 Outline of the Thesis

This thesis is organised as follows:

Chapter 2 presents the background about the data clustering and a brief overview of the different clustering algorithms that are used within different areas. In addition, this chapter discusses two recent trends in clustering, clustering with partial supervision and the parallelisation of clustering algorithms.

Chapter 3 presents new measures - called clustering performance measures (*CPMs*) - for testing the reliability of clustering algorithms. These *CPMs* are defined using a *validation measure*, which determines how well the algorithm works with a given set of parameter values, and a *repeatability measure*, which is used for studying the stability of the clustering results and has the ability to estimate the correct number of clusters in a dataset. In addition, a novel cluster validity index, V_I index, which is able to handle non-spherical clusters, is proposed.

Chapter 4 proposes a novel clustering algorithm, which is a RAdius based Clustering ALgorithm (*RACAL*) and does not require the knowledge of the number of clusters. The proposed algorithm is enhanced by a reliable validity index to choose the best clustering results for a given input parameter. Additionally, an adaptive partial supervision strategy is proposed for using in conjunction with *RACAL* to make it act as a classifier. Furthermore, a parallel algorithm for *RACAL* is proposed to reduce the clustering time of large datasets.

Chapter 5 proposes a novel clustering algorithm, which is called Nearest Neighbour Clustering Algorithm (*NNCA*) and uses the same concept as the K-nearest neighbour (*KNN*) classifier with the advantage that the algorithm needs no training set and it is completely unsupervised. Furthermore, it achieves clustering without any control of cluster sizes. Additionally, a partial supervision strategy is proposed for use in conjunction with *NNCA* to make it act as a classifier. Furthermore, a parallel algorithm for *NNCA* is proposed to reduce the clustering time of large datasets.

Chapter 6 investigates the use of a recently developed clustering algorithm, Self-Organising Oscillator Network (*SOON*) that has biological roots and does not require the knowledge of the number of clusters, in analysing microarray data. In addition, a novel

integrated clustering performance measure (*ICPM*) for assessing the reliability of results from a clustering algorithm used for analysing microarray data is proposed.

Chapter 7 presents an overview of the work done in this research, conclusions and further work.

1.4 Publications

Journal Papers

1. S.A. Salem, N.M. Salem, and A.K Nandi, "Segmentation of Retinal Blood Vessels using a Novel Clustering Algorithm (RACAL) with a Partial Supervision Strategy," *Medical & Biological Engineering and Computing*, vol. 45, pp. 261-273, March 2007.

2. S.A. Salem, L.B. Jack, and A.K. Nandi, "Investigation of Self Organising Oscillator Networks for use in Clustering Microarray Data,", accepted by *IEEE Transactions on NanoBioscience*.

3. S.A. Salem, N.M. Salem, and A.K. Nandi, "Augmentation of a Nearest Neighbour Clustering Algorithm with a Partial Supervision Strategy for Biomedical Data Classification," accepted for publication as an Invited Paper in the Special Issue on Advances in Medical Decision Support Systems of Expert Systems.

4. S.A. Salem and A.K. Nandi, "Development of Assessment Criteria for Clustering Algorithms," accepted by *Pattern Analysis and Applications*.

5. S.A. Salem and A.K. Nandi, "A Novel Clustering Algorithm (RACAL) and an Adaptive Supervision Strategy for Classification," under revision after receiving review comments from *Pattern Analysis and Applications*: Special Issue on Non-parametric Distance-based Classification Techniques and their Applications

6. S.A. Salem and A.K. Nandi, "Integrated Clustering Performance Measure for Analysing Microarray data," submitted to *IEEE Journal of Selected Topics in Signal Processing*: Special Issue on Genomic and Proteomic Signal Processing.

7. N.M. Salem, S.A. Salem, and A.K. Nandi, "Unsupervised and Single Parameter Retinal Blood Vessels Segmentation using a Vesselness Measure," submitted to *Journal of Computers in Biology and Medicine.*

Conference Papers

1. S.A. Salem, L.B. Jack, and A.K. Nandi, "Improving the initialisation and reliability of the self organising oscillator network," in *Proceedings of the 13th European Signal Processing Conference (EUSIPCO 2005)*, (Antalya, Turkey), 04-08 September 2005.

2. S.A. Salem and A.K. Nandi, "New assessment criteria for clustering algorithms," in *IEEE international workshop in Machine Learning For Signal Processing (MLSP 2005)*, (Mystic, CT, USA), pp. 285-290, 28-30 September 2005.

3. S.A. Salem and A.K. Nandi, "Novel clustering algorithm (RACAL) and a partial supervision strategy for classification," in *IEEE international workshop in Machine Learning For Signal Processing (MLSP 2006)*, (Mynooth, Ireland), pp. 313-318, 06-08 September 2006.

4. S.A. Salem, N.M. Salem, and A.K. Nandi, "Segmentation of retinal blood vessels using a novel clustering algorithm," in *Proceedings of the 14th European Signal Processing Conference (EUSIPCO 2006)*, (Florence, Italy), 04-08 September 2006.

5. S.A. Salem and A.K. Nandi, "Parallel nearest neighbour clustering algorithm (PNNCA) for segmenting retinal blood vessels," in *the 25th IASTED conference on Parallel and Distributed Computing and Networks (PDCN 2007)*, (Innsbruck, Austria), pp. 263-268, 13-15 February 2007.

6. A.K. Nandi and S.A. Salem, "Some issues in clustering gene expression data," in *NCAF workshop*, (Liverpool John Moores University, UK), 22-23 May 2007.

7. N.M. Salem, S.A. Salem, and A.K. Nandi, "Segmentation of retinal blood vessels based on analysis of the Hessian matrix and clustering algorithm," in *Proceedings*

of the 15th European Signal Processing Conference (EUSIPCO 2007), (Poznań, Poland), 03-07 September 2007.

8. S.A. Salem and A.K. Nandi, "A new scalable and efficient parallel algorithm (PRACAL) for clustering large datasets," in *the 19th IASTED conference on Parallel and Distributed Computing and Systems (PDCS 2007)*, (Massachusetts, USA), 19-21 November 2007.

9. S.A. Salem, N.M. Salem, and A.K. Nandi, "Development of a partial supervision strategy to augment a nearest neighbour clustering algorithm for biomedical data classification," in *the international conference on Bio-inspired Systems and Signal Processing (BIOSIGNALS 2008)*. (Accepted)

Chapter 2

Background

2.1 Introduction

THE main focus of this chapter is to give the necessary background about data clustering and some of the recent trends in clustering. This chapter begins with a background about cluster analysis and a brief overview of the different clustering algorithms. Then, the enhancement of the clustering process with partial supervision along with a review of partially (or semi-) supervised clustering algorithms are discussed. Finally, the recent trend in parallel implementations of clustering algorithms, which give a flexible and interactive data exploration when analysing large datasets, is also discussed.

2.2 Cluster Analysis

Cluster analysis is the grouping of individuals in a population in an attempt to discover structures or groups in the data. In some sense, it is desirable for the individuals within a group to be similar to one another, but dissimilar from individuals in other groups [1]. Jain and Dubes [2] defines the cluster analysis as an organisation of the data by abstracting underlying structure either as a grouping of individuals or as a hierarchy of groups. The representation can then be investigated to see if the data group according to preconceived ideas or to suggest new experiments. Many definitions for clustering have been proposed over the years (e.g., [3, 4, 5]). However, a highly descriptive and appealing characterisation

of the clustering or grouping can be found in the literature [6]:

"*Cluster analysis is the art of finding groups in data*"

Clustering has been applied in a wide variety of fields [7, 8], ranging from engineering (machine learning, artificial intelligence, pattern recognition, mechanical engineering, electrical engineering), computer sciences (web mining, spatial database analysis, textual document collection, image segmentation), life and medical sciences (genetics, biology, microbiology, paleontology, psychiatry, clinic, pathology), to earth sciences (geography, geology, remote sensing), social sciences (sociology, psychology, archeology, education), and economics (marketing, business). Accordingly, clustering is also known as numerical taxonomy, learning without a teacher (or unsupervised learning), typological analysis and partition. The diversity reflects the important position of clustering in scientific research. On the other hand, it causes confusion, due to the differing terminologies and goals. Clustering algorithms developed to solve a particular problem, in a specialised field, usually make assumptions in favour of the application of interest. These biases inevitably affect the performance in other problems that do not satisfy these premises. In Section 2.2.3, categorisation of clustering algorithms will be discussed.

2.2.1 Definitions

- An *object* O_i, a pattern (or a feature vector), is a single data point used by a clustering algorithm [9].
- A *feature* (or *attribute*) is an individual component of an object [9].
- A *dataset* O is a set of objects. In many cases, a dataset is viewed as an $n \times d$ matrix (n objects each of d features).
- A *cluster* is a set of similar objects, and objects from different clusters are not similar.
- *Hard* (or *Crisp*) clustering algorithms assign each object to one and only one cluster.
- *Fuzzy* clustering algorithms assign to each object a membership degree to each cluster. The membership degrees are recorded a partition matrix, $U(O) = [u_{ij}]_{K \times n}$,

where K is the number of clusters, and n is the number of objects.

- A *distance measure* is a metric used to evaluate the similarity of objects [9].
- A *Prototype* is a cluster centre.

2.2.2 Basic clustering step

2.2.2.1 Preprocessing

Most clustering methods assume that d-dimensional feature vectors represent all data objects. This step therefore involves choosing an appropriate feature, and doing appropriate preprocessing and feature extraction on data objects to measure the values of the chosen feature set. It will often be desirable to choose a subset from all the features available, to reduce the dimensionality of the problem space. More information on feature selection can be found in [1, 10].

2.2.2.2 Similarity measure

Similarity measure plays an important role in clustering process. The notion of similarity or resemblance, distance between two objects is a dominant factor of any grouping [11]. It is so common that we tend to reach some simplification of the problem. The role of the distance function is to quantify a notion of similarity; the lower the distance between two objects, the higher the level of their similarity. This property implies that any two objects with the distance function equal to 0 are the same and therefore similar to the degree of 1. Euclidean distance is the most widely used distance function, and can be defined as

$$D_2(O_i, O_j) = (\sum_{k=1}^{d}(O_{i,k} - O_{j,k})^2)^{1/2} \tag{2.1}$$

which is a special case ($q = 2$) of the Minkowski distance [9] which is defined as

$$D_q(O_i, O_j) = (\sum_{k=1}^{d}(O_{i,k} - O_{j,k})^q)^{1/q} \tag{2.2}$$

The drawback of Minkowski distance is the tendency of the largest-scaled feature to dominate other features. This can be solved by normalising the features to a common range [9]. One way to do this is by using cosine distance (or vector product). When $q = 1$, the Minkowski distance is referred as Manhattan distance [1, 7]. The Manhattan distance is usually not suitable for clustering data of high dimension, because the Manhattan distance between objects increases with the increase in dimensionality. Linear correlation among features can also distort distance measures; this distortion can be alleviated by applying Mahalanobis distance [12] which is defined as:

$$D_M(O_i, O_j) = (O_i - O_j)\sum^{-1}(O_i - O_j)^T \tag{2.3}$$

where $\sum$ is the covariance matrix of the objects. The Mahalanobis distance gives different features different weights based on their variances and pairwise linear correlations. Furthermore, it implicitly assumes that the densities of the classes are multivariate Gaussian [9]. There are other distance measures and can be found in [5, 13, 14].

2.2.2.3 Clustering algorithms

Clustering algorithms are general methods, which use particular similarity measures. The particular choice of clustering algorithms depends on the desired properties of the final clustering, e.g. compactness, and connectedness. Once a similarity measure is chosen, the construction of a clustering algorithm function makes the partition of clusters an optimisation problem, which is well defined mathematically, and has rich solutions in the literature [8].

2.2.2.4 Cluster validation

Given a dataset, each clustering algorithm can always produce some clusters regardless of whether or not clusters exist. Moreover, different approaches usually lead to different clusters; and even for the same algorithm, parameter identification or the presentation order of the input data objects may affect the final results. Therefore, effective evaluation standards and criteria are important to provide the users with a degree of confidence for the clustering

results derived from the used algorithms. These assessments should be objective and have no preferences to any algorithm. Also, these should be useful for answering questions like how many clusters are hidden in the data, whether the clusters obtained are meaningful or just an artifact of the algorithms, or why one would choose an algorithm instead of another.

2.2.2.5 Result interpretation

The ultimate goal of clustering is to provide users with meaningful insights from the original data, so they can effectively solve the problems encountered. In data compression applications, one can represent similar objects with fewer objects, cluster representatives. In model construction application, one can build a model of the problem based on the clusters formed from the data.

2.2.3 Overview of clustering algorithms

The idea of data grouping, or clustering, is simple in its nature and is one of the most primitive mental activities of humans [7]; whenever we are presented with a large amount of data, we usually tend to summarise this huge number of data into a small number of groups (clusters) in order to facilitate its analysis. Moreover, most of the data collected in many problems seem to have some inherent properties that lend themselves to natural groupings. Nevertheless, finding these clusters is not a simple task for humans unless the data is of low dimensionality (two or three dimensions). This is why some algorithms have been proposed to solve this kind of problem. Those algorithms are called "Data Clustering Algorithms".

There are literally many clustering algorithms that are well-reported in the literature and illustrated with carefully selected benchmarks. Obviously, it is impossible to discuss all of them. It is, however, worth establishing a general taxonomy of clustering algorithms. In general, there are two basic types of clustering algorithms: *Hierarchical* and *Partitional* algorithms. These algorithms are classified based on the properties of the generated clusters [1, 5, 7, 9, 11].

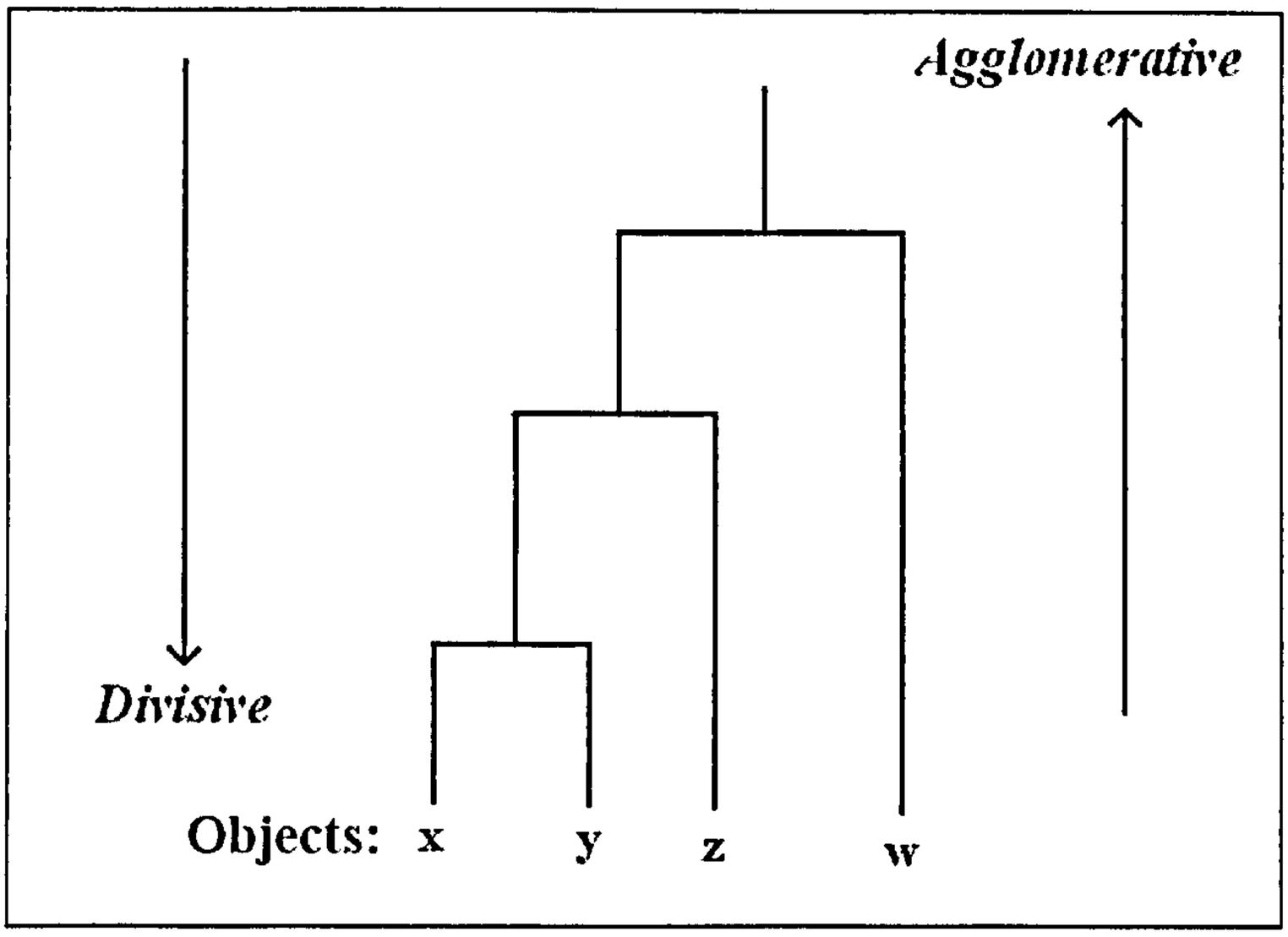


Figure 2.1: A sample dendrogram of clustered data.

- ***Hierarchical*** algorithms create a hierarchical decomposition of the data. The hierarchical decomposition is represented by a tree structure, binary tree or dendrogram, that splits the data into small subsets until each subset consists of only one object. In such a hierarchy, each node of the tree represents a cluster of data. The tree can be built either from the leaves up to the root by merging clusters (agglomerative clustering) or from the root down to the leaves by dividing clusters at each step (divisive clustering). It is necessary that a termination condition is defined to indicate when the merge or division process should stop.

  - Divisive: Start with one cluster of all objects and successively split a cluster until only singleton clusters of individual points remain. For a cluster with n objects, there are $2^{n-1} - 1$ possible two-subset divisions, which is very expensive in computation [5]. Examples of divisive clustering algorithms, MONA and DIANA, are described in [6].

  - Agglomerative: Start with each object being as individual cluster, and successively merge the most similar or closest pair of clusters according to a distance measure.

Figure 2.1 depicts the dendrogram produced by either a divisive or agglomerative clustering algorithms.

Based on the definition of the distance measure between clusters, there are many agglomerative clustering algorithms. The most popular methods include single linkage, complete linkage and average linkage methods [11].

- *Single linkage:* The similarity between two clusters, S and T, is calculated based on the minimal distance between the objects belonging to the corresponding clusters.

$$\|T - S\| = \min_{x \in T, y \in S} \| x - y \| \tag{2.4}$$

- *Complete linkage:* The similarity between two clusters, S and T, is calculated based on the maximal distance between the objects belonging to the corresponding clusters.

$$\|T - S\| = \max_{x \in T, y \in S} \| x - y \| \tag{2.5}$$

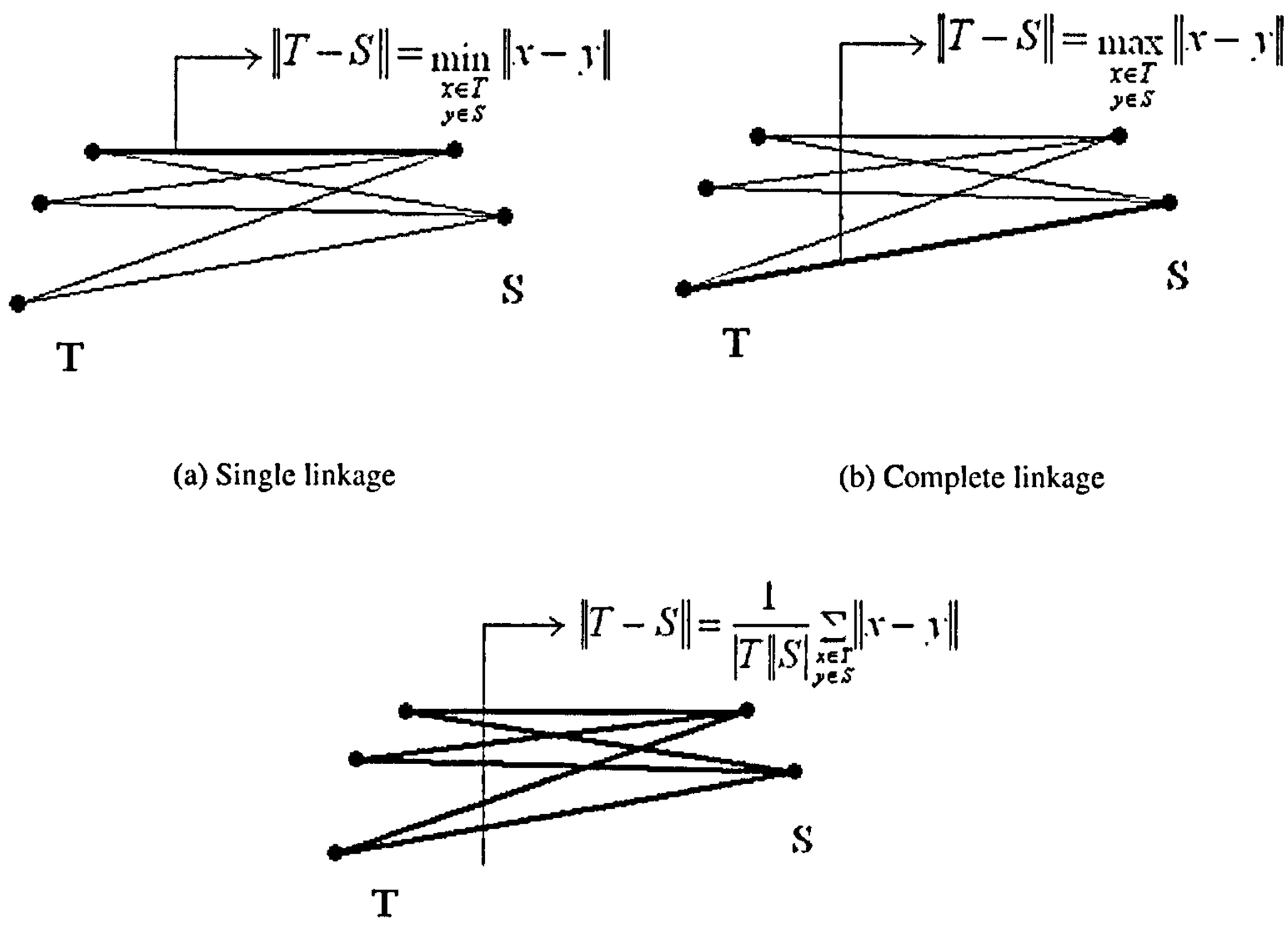
- *Average linkage:* The similarity between two clusters, S and T, is calculated based on the average distance between all possible pairs of objects in the clusters.

$$\|T - S\| = \frac{1}{|T|\,|S|} \sum_{x \in T, y \in S} \| x - y \| \tag{2.6}$$

where $|T|$ and $|S|$ denote the number of objects in clusters T and S respectively.

These three standard methods of constructing clusters are visualised in Fig. 2.2.

The common criticism for hierarchical clustering algorithms comes from the fact that once a merge or a split is committed, it cannot be undone or refined, which means that hierarchical clustering algorithms are not capable of correcting possible previous misclustering. Furthermore, the computational complexity of hierarchical clustering algorithms is at least $O(n^2)$ and this high cost limits their application in large datasets [8, 11]. Recently, many hierarchical clustering algorithms have been



(a) Single linkage (b) Complete linkage

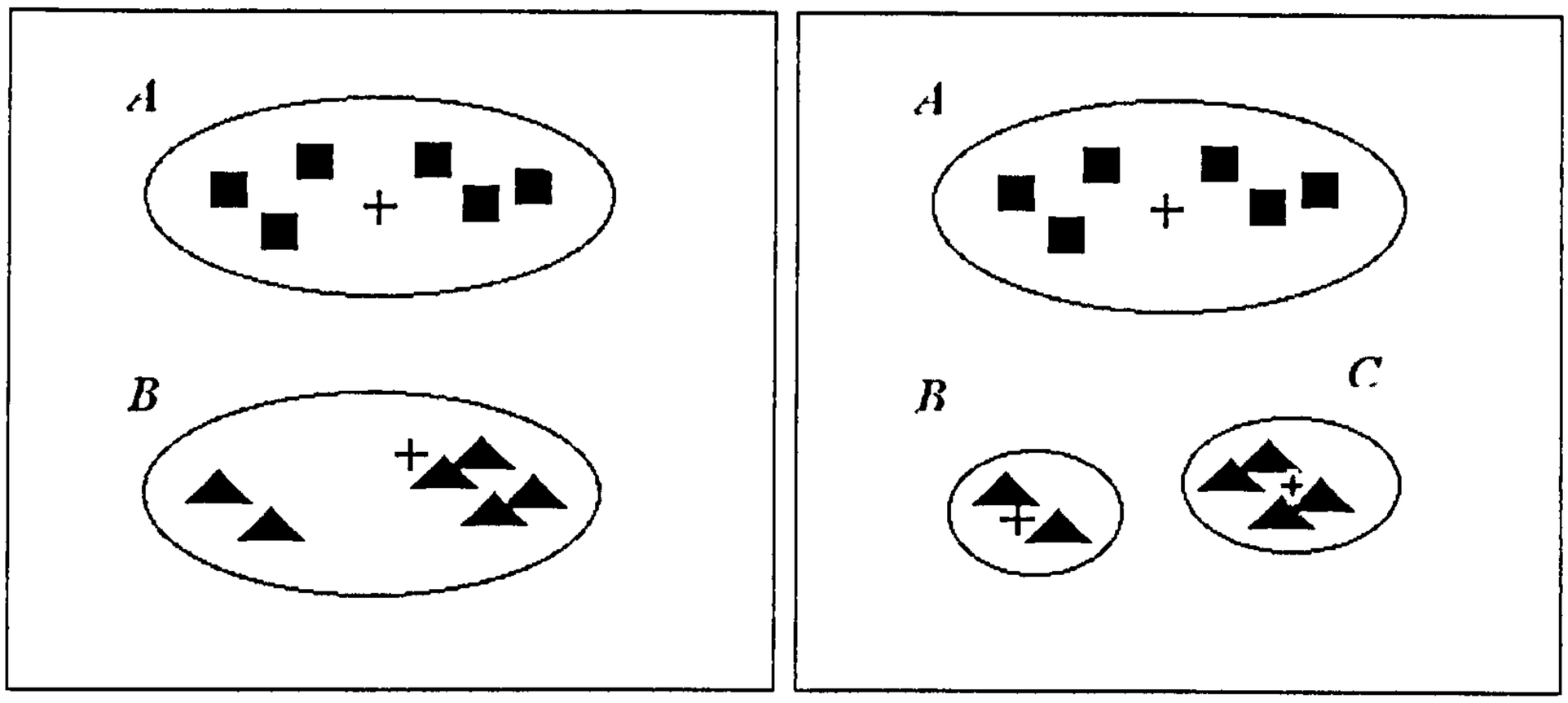
(c) Average linkage

Figure 2.2: Constructing clusters with the use of hierarchical methods.

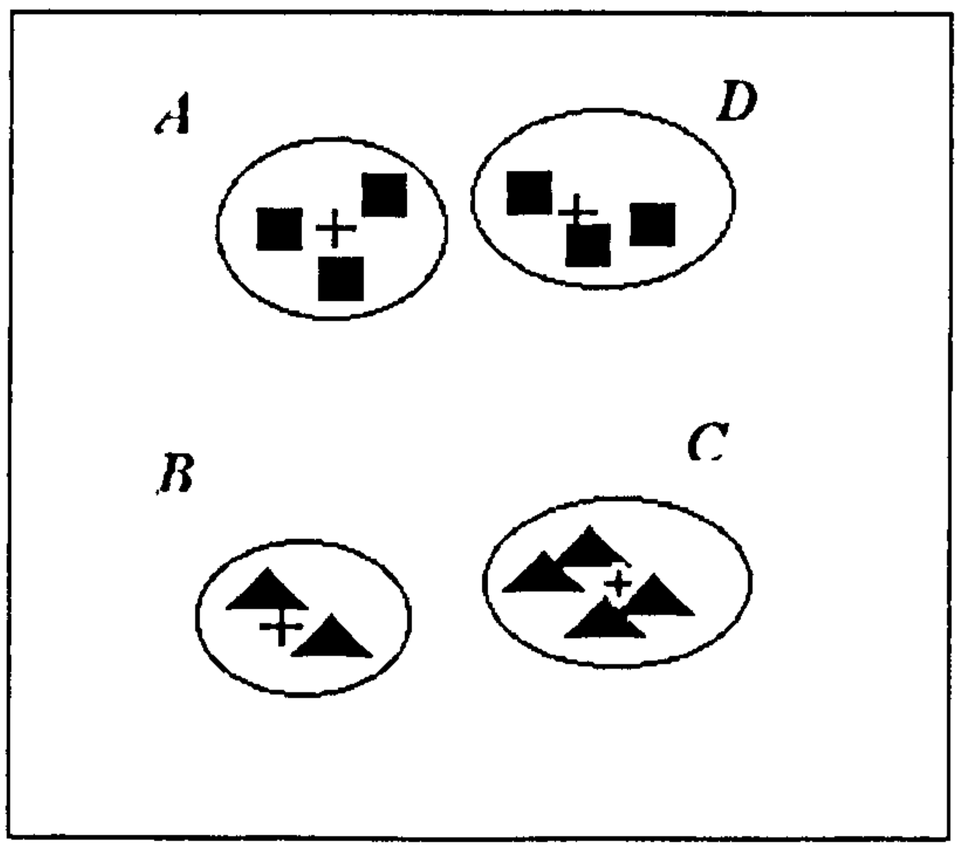
proposed to improve the clustering performance and to handle large datasets, e.g., BIRCH [15], CURE [16], and CHAMELEON [17].

- ***Partitional*** algorithms construct a partition of the data of n objects into a set of K clusters where K is an input parameter. Typically, partitioning algorithms start with an initial partition of the data and then use an iterative control strategy to optimise the clustering quality, e.g. the average distance of an object to its representative. Figure 2.3 shows an example of partitional clustering at different number of clusters. The most commonly used partitional clustering algorithms are K-means [2, 6], K-medoids [6], CLARA (Clustering Large Applications based on a Randomised Search) [6], and CLARANS (Clustering LARge ApplicatioNs) [18].

  - *K-means* clustering algorithm is one of the best-known and most commonly used clustering algorithms [1]. Its goal is to produce K clusters from a set



(a) (b)



(c)

Figure 2.3: Examples of partitional clustering at different number of clusters (K): Clustering with (a) $K = 2$, (b) $K = 3$, and (c) $K = 4$.

of n objects (each object is characterised by d features) so as to minimise the within-cluster discrepancies, where a discrepancy is defined as the difference between an object and the cluster centre. The standard measure of discrepancy used is the L_2 norm. This leads to the following objective function:

$$E = \sum_{j=1}^{K} \sum_{x_i \in C_j} \|x_i - C_j\|^2 \tag{2.7}$$

where the index j refers to the cluster label, C_j is the centre of cluster j and x_i refers to the objects to be clustered. The algorithm is as follows:

1- Select K objects as initial centres.

2- Assign each data object to the closest centre.

3- Recalculate the centres of each cluster.

4- Repeat step 2 and 3 until centres do not change.

A great advantage of the K-means algorithm is that its time complexity is $O(JKn)$ with J being the number of iterations, making it slightly more scalable. However it is highly sensitive to noise and outliers, and it requires the user to specify the number of clusters in advance. Furthermore, the definition of "means" limits the application to numerical variables.

- *K-medoids* clustering algorithm [6] is an extension of K-means, intended to handle outliers efficiently. Instead of cluster centres, it chooses to represent each cluster by its medoids. A medoid is the most centrally located object inside a cluster. As a consequence, medoids are less influenced by extreme values; the mean of a number of objects would have to take account of these values while a medoid would not. The algorithm chooses K medoids initially and tries to place other objects in clusters whose medoid is closer to them, while it swaps medoids with non-medoids as long as the quality of the result is improved. Quality is also measured using squared-error between the objects in a cluster and its medoid. The computational complexity of K-medoids is $O(JK(n-K)^2)$ with J being the number of iterations, making it very costly

for large K and $(n - K)$ values. This high cost limits its application to cluster large datasets. CLARA and CLARANS algorithms [6, 18] have been proposed to extend the K-medoids algorithm for tackling large applications.

For hard partitional clustering, each object only belongs to one cluster. Fuzzy clustering extends this notion to associate each object to all clusters with a degree of membership, $u_{ij} \in [0, 1]$, which represents the membership degree of j-th object to i-th cluster. Larger membership values indicate higher confidence in the assignment of the object to the cluster. One widely used algorithm is the Fuzzy C-Means (FCM) algorithm [19, 20], which is based on K-means clustering algorithm.

Recently, there are several advances on K-means and other partitional clustering algorithms with their applications which can be found in [21, 22, 23].

Apart from the two main categories of partitional and hierarchical clustering algorithms, many other methods have emerged in cluster analysis, and are mainly focused on specific problems or specific datasets available. These methods include [13]:

- **Density-Based Clustering**: These algorithms cluster objects according to specific density objective functions. Density is usually defined as the number of objects in a particular neighbourhood of a data objects. In these approaches a given cluster continues growing as long as the number of objects in the neighbourhood exceeds some parameter. This is considered to be different from the idea in partitional algorithms that use iterative relocation of points given a certain number of clusters. Examples of density-based clustering algorithms are: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm [24], which is the most well known, and DENCLUE (DENsity-based CLUstEring) algorithm [25].

- **Grid-Based Clustering**: The main focus of these algorithms is spatial data, i.e., data that model the geometric structure of objects in space, their relationships, properties and operations. The objective of these algorithms is to quantise the dataset into a number of cells and then work with objects belonging to these cells. These algorithms do not relocate objects but rather build several hierarchical levels of groups

of objects. In this sense, these algorithms are closer to hierarchical algorithms but the merging of grids, and consequently clusters, does not depend on a distance measure but it is decided by a predefined parameter. Examples of grid-based clustering algorithms are: STING (STatistical INformation Grid) [26], and WaveCluster [27] algorithms.

- **Model-Based Clustering**: These algorithms find good approximations of model parameters that best fit the data. These algorithms can be either partitional or hierarchical, depending on the structure or model hypothesised about the dataset and the used method to refine this model to identify partitionings. These algorithms are closer to density-based algorithms, in that particular clusters are grown so that the preconceived model is improved. However, these algorithms sometimes start with a fixed number of clusters and do not use the same concept of density. Examples of model-based clustering algorithms are: EM (Expectation-Maximization) algorithm [28] and the Kohonen Self Organising Map (SOM) that has been perhaps one of the most popular unsupervised clustering algorithms and is used in many different applications [29].

- **Graph-Based Clustering**: These algorithms uses the concepts and properties of graph theory, where the nodes N of a weighted graph G are corresponding to the data objects and the edges E reflect the distances between each pair of data objects. Examples of graph-based clustering algorithms are: spectral and path based clustering algorithms [30, 31]. These algorithms consider arbitrarily shaped clusters which are non-linearly separable in feature space.

- **Categorical Data Clustering**: These algorithms are specificially developed for data where Euclidean, or other distance measures cannot be applied. Examples of such algorithms can be found in [32, 33]

Finally, one can conclude that there is no best clustering algorithm. Different clustering methods can yield different results and some methods will fail to discover obvious clusters. The reason for this is that each method implicitly forces a structure on the given data. For

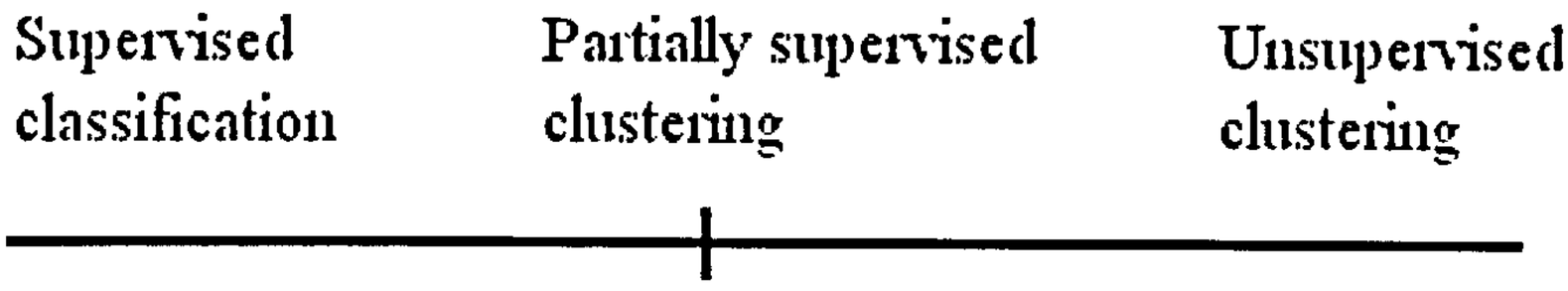


Figure 2.4: Clustering with respect to knowledge.

example, the sum-of-squares methods, e.g., K-means, will tend to produce hyperspherical clusters. Therefore, there is no clustering algorithm that can be universally used to solve all problems.

2.3 Clustering with Partial Supervision

There are two fundamental modes of learning [11, 13, 34]:

- Supervised learning
- Unsupervised learning

In *supervised* learning, the process of assigning data objects to groups is known as classification. This process relies on the availability of knowledge about the data being analysed. Knowledge here represents the set of labels associated with the data. In supervised mode, the class label and the number of classes are predefined.

In *unsupervised* learning, the process of assigning unlabelled data objects to clusters using some similarity measure (i.e. distance-based, density-based, etc.) is known as clustering. This process is self-supervised. Ideally, two criteria have to be satisfied, namely intra-cluster similarity and inter-cluster dissimilarity.

These modes appear to be extreme in the sense that the supervised mode requires complete knowledge of the data while the unsupervised uses no knowledge. Figure 2.4 shows

the clustering with respect to knowledge. The complete knowledge of the data is often limited and expensive to generate, since labeling data typically requires human expertise. Therefore, the main idea of clustering with partial supervision, semi-supervised clustering, is to take the advantage of the smaller proportion of labelled objects to guide the clustering process of the unlabelled objects. Pedryez and Waletzky [35] define clustering with partial supervision as phenomenon occurred when in addition to a vast number of unlabelled objects one is also furnished with some (usually, few) labelled objects. Definitely, these few already classified objects, when carefully exploited, could provide some general guidance to the clustering mechanism. As indicated in [36], the labelled objects serve as "anchor" (reference) elements that shape the clusters.

One of the typical applications of clustering with partial supervision is Computer-Aided Diagnosis (CAD) which has become one of the major research subjects in medical imaging and diagnostic radiology [37]. The basic concept of CAD is to provide a computer output as a second opinion to assist radiologists' image interpretation by improving the accuracy and consistency of radiological diagnosis [37]. The design of clustering with partial supervision in CAD can play important role in improving CAD performance with small amount of knowledge, where only some labelled objects or regions of an image can assist in identification of any suspicious objects or regions.

Several partially (or semi-) supervised clustering algorithms have been proposed. For example, Pedryez and Waletzky [35] proposed a modified version of the standard FUZZY ISODATA algorithm to deal with the problem of partial supervision. In this work, the classification information is incorporated additively as a part of an objective function utilised in the standard FUZZY ISODATA algorithm.

Basu *et al.* [38] proposed two semi-supervised algorithms based on a seeding mechanism. These algorithms rely on the K-means algorithm. In the first algorithm, the seeds are used to initialise the partition centres and then updated during the clustering process. In the second algorithm, once initialised with the seeds, the centres are not updated. The idea here is that when seeds are noise-free, centres may be kept unchanged.

Blum *et al.* [39], and Zhu *et al.* [40] consider the data (labelled and unlabelled) as a graph. The data objects are represented as nodes which are interconnected by weighted edges. The weights indicate the similarity between objects. Basically, these approaches use a loss function and a regularisation factor to propagate labels of the labelled objects to the unlabelled objects lying in the vicinity.

Demiriz *et al.* [41] applied genetic algorithms to combine supervised classification and unsupervised clustering. The basic idea in this work is to minimise an objective function that is a linear combination of the cluster dispersion and the cluster impurity of the form:

$$a \times Cluster\ Dispersion + b \times Cluster\ Impurity \tag{2.8}$$

The first component of this formula is concerned with unsupervised clustering while the second component controls the purity of the generated clusters and is therefore concerned with supervised classification. If $a = 0$, then the result is purely unsupervised clustering algorithm. If $b = 0$, the result is a purely supervised algorithm.

Jeon and Landgrebe [42] suggested a partially-supervised classification algorithm to discriminate a particular class of interest. The goal was to design a classification algorithm given only the labelled data objects. The proposed algorithm relies on three steps. In the first step, each data object is assigned a weight which represents the likelihood of not being in the class of interest. In the second step, the clusters are initialised using a probabilistic unsupervised algorithm. In the last step, clusters are refined and adjusted.

Basu *et al.* [43] proposed a probabilistic model for semi-supervised clustering based on hidden markov random fields that incorporates supervision into prototype-based clustering.

Nigam *et al.* [44] investigated a probabilistic approach for text classification. The approach combines the Expectation-Maximization (EM) algorithm and a naive Bayes classifier. The algorithm trains the classifier using the labelled data only. Then, the labels of the unlabelled objects are iteratively estimated and the classifier is re-trained using all labelled data until convergence.

2.4 Parallel Data Clustering

Many algorithms for data clustering developed in recent decades all face a major challenging problem in scaling up to very large datasets, where the computational cost of such algorithms increases with the increase in data sizes and dimensions. Although, many attempts have been proposed to improve the performance of some conventional clustering algorithms, these attempts are facing more severe challenges in handling the rapid growth of the data than the conventional algorithms. For example, K-medoids clustering algorithm is designed for tackling not only the effect of noise and outliers in the data, which are worst affecting the performance of K-means clustering algorithm, but also the memory required to store a vast amount of information for computing the means. However, K-medoids is not as efficient computationally as K-means, as its computational time is higher than K-means algorithm. One possible solution such as sampling the data can be used, however the data sampling approaches are not always accurate enough for discovering representative patterns (objects), e.g. CLARA algorithm uses random sampling approach to tackle the computational required by K-medoids algorithm.

In general, clustering algorithms are very computionally demanding and, thus, require high-performance machines to get results in a reasonable amount of time. Experiences of clustering algorithms taking one week or about 20 days of computation time on sequential machines are not rare [45]. One way of overcoming this limitation is to improve the operating speed of processors and other components so that they can offer the computational power required by a clustering algorithm. Even though this is currently possible to a certain extent, future improvements are constraints by the speed of light, thermodynamics laws, and the high financial costs for processor fabrications. A viable and cost-effective alternative solution is to use scalable parallel computers that can provide the appropriate setting to efficiently execute clustering algorithms for analysing large datasets.

Recently, there is an increasing interest in parallel implementations of data clustering algorithms. This reduces the execution time of the clustering process which results in a flexible and interactive data exploration. Examples of parallel approaches to clustering can be found in [46, 47, 48, 49, 50, 51, 52, 53].

2.5 Summary

In this chapter, the necessary background about the data clustering and a brief overview of different clustering algorithms have been introduced. The second part of this chapter discussed the advantage of enhancing the clustering algorithms with partial supervision to improve the clustering performance. Finally, the recent trend in parallelising clustering algorithms to handle large datasets has been discussed.

Chapter 3

Clustering Performance Measures

3.1 Introduction

CLUSTERING is an unsupervised classification, and is usually more difficult to assess than a supervised approach. The procedure for evaluating the results of a clustering algorithm is known as *cluster validation*. In general terms, there are three approaches to investigate cluster validity [7, 54]. The first approach is based on *external criteria*. This implies that results of a clustering algorithm are evaluated based on a pre-specified structure, which is imposed on a dataset and reflects the user's intuition about the clustering structure of the dataset. The second approach is based on *internal criteria*. In this case clustering results are evaluated in terms of quantities that involve vectors of the dataset themselves (e.g. proximity matrix). The third approach to cluster validity is based on *relative criteria*. Here the basic idea is to evaluate the clustering structure by comparing it to other clustering results using the same algorithm, but with different input parameter values. The cluster validity approaches based on external and internal criteria rely on statistical hypothesis testing. Moreover, the indices related to these approaches aim at measuring the degree to which a dataset confirms to an *a-priori* specified scheme [54]. On the other hand, the third approach aims at finding the best clustering scheme that a clustering algorithm can define under certain assumptions and parameters [55]. There are many validity indices which are used for relative criteria, namely, the Davies-Bouldin index [56], Dunn's index

[57], Calinski-Harabasz index [58], "I" index [59], and many other indices described in [55, 60, 61, 62, 63, 64].

The major drawback of external and internal criteria is their high computational cost [65]. Moreover, these are used to decide whether an algorithm performs its function or not, and cannot tell how well an algorithm works for the data that contain overlapped clusters as well as well-separated data. In relative criteria, the evaluation of the clustering scheme and determining the optimum parameters ignores the effect of the initialisation and randomness on the part of the algorithm, i.e., one can get different results for two different runs on the same dataset using the same parameters, and this can pose extra problems for the clustering algorithm using higher dimensional data that contains overlapped structures. Maulik and Bandyopadhyay [59] initialised the same set of cluster centres in order to make comparison among clustering algorithms fair, but the main drawback of this approach is that it does not study the effect of different initialisation on the degree of stability and the evaluation procedure to get the optimum setting of the parameters.

The problem of deciding the number of clusters in the data is common to all clustering methods [1]. There are numerous algorithms for cluster validity in the literature. These methods use some criteria to estimate the number of clusters that are present in the dataset. Tibshirani *et al.* [66] propose the Gap statistic for estimating the number of clusters in a dataset, it relies on computing the total sum of within-cluster dissimilarities on the dataset for a given number of clusters which corresponds to the squared-error criterion optimised by the K-means algorithm [2]. Then, this quantity is compared against the average over data which is uniformly sampled from a hyper-rectangle containing the original data. The main drawback of Gap statistic is the bias to spherical distribution datasets with compact clusters packed around cluster centres. Law *et al.* [67] propose bootstrapping approach for estimating the number of clusters by considering a clustering algorithm as an estimator for the partition of the data space. It uses bootstrapping to estimate the variability of this estimator. If the partition is valid, the variability should be low. Lange *et al.* [68] propose stability-based model selection for estimating the number of clusters by considering the average dissimilarity of solutions computed on two different datasets that have been

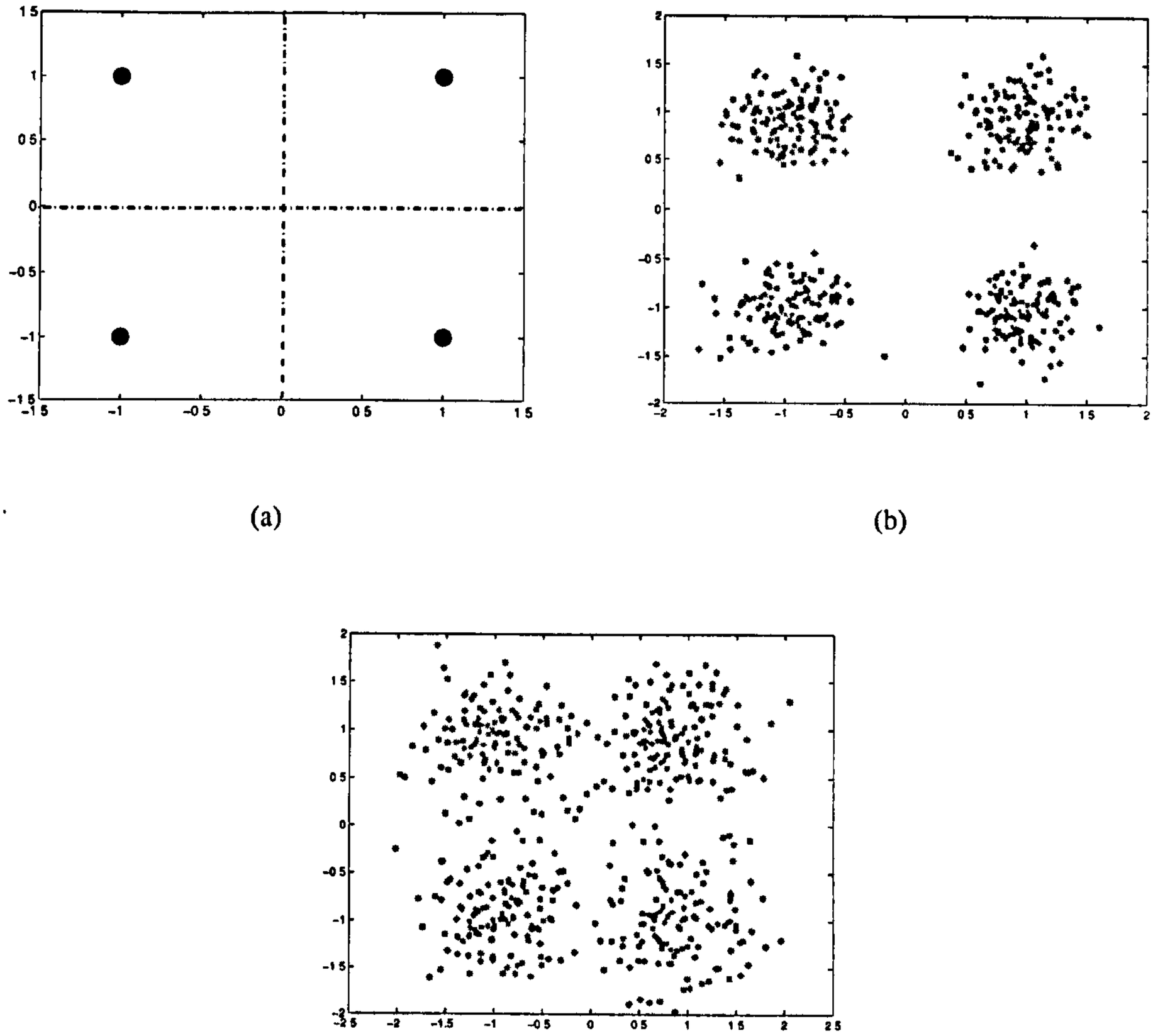
generated by the same source. The estimated number of clusters corresponds to the minimum dissimilarity between partitions for different number of clusters. Other approaches [69, 70] cluster two nondisjoint datasets to measure the similarity of the clustering structures obtained for the intersection of both datasets and many other methods described in [60, 71, 72, 73, 74].

In the literature, numerous approaches have been proposed to assess clustering results. These approaches are belong to either validity based approach or stability based approach. The validity based approach is used to assess the quality of clustering results, while stability based approach is used to assess the stability of clustering results. Therefore, there is a need for performance measures that are able to examine not only the quality of the clustering results but also the stability of results in a quantitive manner. This offers guidelines for choosing the proper clustering algorithm for a particular dataset, and the parameter settings which give the most promising results.

In this chapter, new measures - called clustering performance measures (CPMs) - for testing the robustness and reliability of a clustering algorithm are introduced. The proposed CPMs can be used to evaluate different clustering algorithms. These measures are applied on sub-sampled datasets, the original dataset, and datasets with extra samples. Therefore, one can use the proposed CPMs to evaluate: 1) clustering algorithms that have a structure bias to a certain type of data distribution as well as those that have no such biases, 2) clustering algorithms that have initialisation dependency (e.g. K-means algorithm) as well as the clustering algorithms that have a unique solution for a given set of parameter values with no initialisation dependency (e.g. hierarchical algorithm [1, 7]). Furthermore, a novel cluster validity index, which is called V_I index, and a repeatability measure as alternative methods for estimating the number of clusters in a dataset are proposed.

3.2 Datasets

In order to examine the CPMs to be proposed in Section 3.3, different types of real-world and synthetic datasets are used.

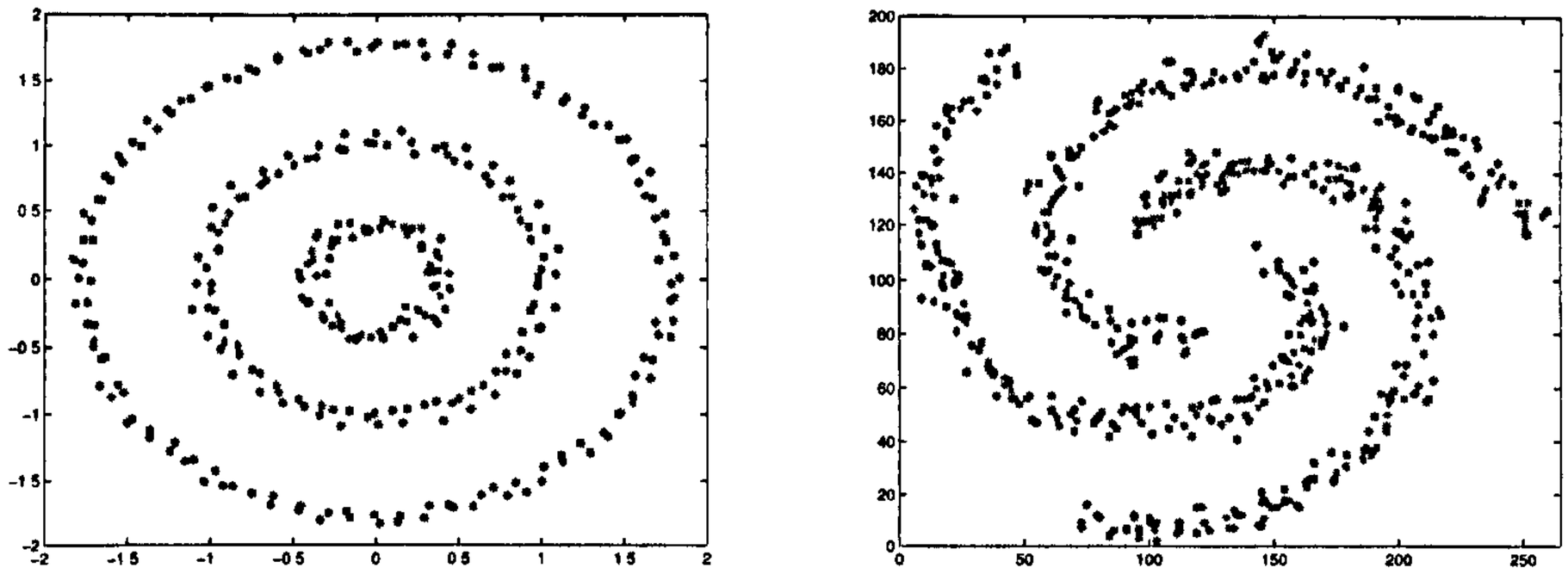


(a) (b) (c)

Figure 3.1: (a) Constellation diagram of communications data, (b) QAM-4 at SNR = 15 dB, and (c) QAM-4 at SNR = 10 dB.

3.2.1 Communications data

Communications data are good benchmark test problems for unsupervised clustering algorithms, primarily due to clean delineation between different clusters within constellation [75] at high SNR (Signal to Noise Ratio). However, by lowering the SNR of the signal, less well separated clusters result, which can pose more of a problem to a clustering algorithm. Figure 3.1 shows the constellation diagram of communications data and two QAM-4 datasets with different SNR values.



(a) (b)

Figure 3.2: Synthetic datasets containing (a) three ring-shaped clusters, and (b) spiral arms.

3.2.2 Breast cancer data

Two Wisconsin breast cancer datasets [76] are considered in this chapter. The first contains 683 samples of 9 features each, and two classes: Benign (class 1 and 444 samples) and Malignant (class 2 and 239 samples). The second contains 569 samples of 30 features each, and two classes: Benign (class 1 and 357 samples) and Malignant (class 2 and 212 samples).

3.2.3 Synthetic data

Two synthetic datasets are used to evaluate the clustering algorithms that are able to cluster datasets with arbitrarily shaped clusters. Figure 3.2 shows two synthetic datasets.

3.3 Clustering Performance Measures - basis and definitions

For reliable and fair evaluation, two stages are used to measure the robustness and reliability of a clustering algorithm; The first is the *validation measure* which is used for determining how well the algorithm works for a given set of parameter values. The second is the *repeatability measure* which is used for studying the stability of the clustering solutions. The proposed CPMs are derived from these two measures.

3.3.1 Validation measure

The assessment of most clustering algorithms using visual tools is a crucial verification of the clustering results; however it is very difficult for humans that are not accustomed to higher dimensional space. Therefore, cluster validity methods are used for evaluating and assessing results from a clustering algorithm. Consequently, the efficiency and degree of robustness of a clustering algorithm can be determined by reliable validity indices to judge the relative merits of clustering structures in a quantitative manner, i.e. a good clustering algorithm is the one that can achieve better validation values for many possible sets of parameter values. It should be noted that some clustering algorithms are biased to certain types of data distribution (e.g., K-means algorithm has a structure bias to spherical distribution clusters). Other types of clustering algorithms have no such biases (e.g., spectral clustering algorithm [77]).

In case of no structure biases

A new validation index, which is called V_I index, to assess clustering results of clustering algorithms that are able to cluster the datasets with arbitrarily shaped clusters (e.g. spectral clustering algorithm [77]) is proposed. The proposed V_I index is based on a connectivity criterion which helps in determining the *intercluster* separation and the *intracluster* scatter that respectively corresponding to the *between-cluster* separation and *within-cluster* scatter of the validity indices that have a structure bias to compact or spherically distributed clusters. The intracluster scatter and intercluster separation are based on the creation of a minimum spanning tree [78] which is a graph with no cycles. It minimises the total length over all possible spanning trees. In clustering, a minimum spanning tree represents the data objects as nodes which are connected by weighted edges. The weights indicate the similarity between objects.

The intracluster scatter of the i-th cluster, Ia_i, is equal to the largest dissimilarity of the minimum spanning tree [78], T_i, that connects the memberships of cluster i, i.e., $Ia_i = \max_{a,b \in T_i} d_{T_i}(x_a, x_b)$, where $d_{T_i}(x_a, x_b) = \| x_a - x_b \|$ is the dissimilarity between two consecutive objects in T_i. Small values of Ia_i refer to closer memberships. The intercluster

separation between cluster i and cluster j, Ie_{ij}, is defined as the maximum dissimilarity between two consecutive objects in the minimum spanning tree (T_{ij}) when cluster i and cluster j are merged together, i.e., $Ie_{ij} = \max_{a,b \in T_{ij}} d_{T_{ij}}(x_a, x_b)$. The overall intercluster separation of cluster i, Ie_i, is equal to the minimum intercluster separation between cluster i and all other clusters, i.e., $Ie_i = \min_{j=1, j \neq i}^{K} Ie_{ij}$. Large values of Ie_i refer to large cluster separations. The V_I index is defined as the ratio of the sum of intercluster separation to the sum of intracluster separation, as described in Eq. 3.1. Large values of V_I index indicate the presence of low intracluster and high intercluster separations. Furthermore, the number of clusters that maximises the V_I index is considered as the optimal number of clusters.

$$V_I(K) = \frac{\sum_{i=1}^{K} Ie_i}{\sum_{i=1}^{K} Ia_i} \tag{3.1}$$

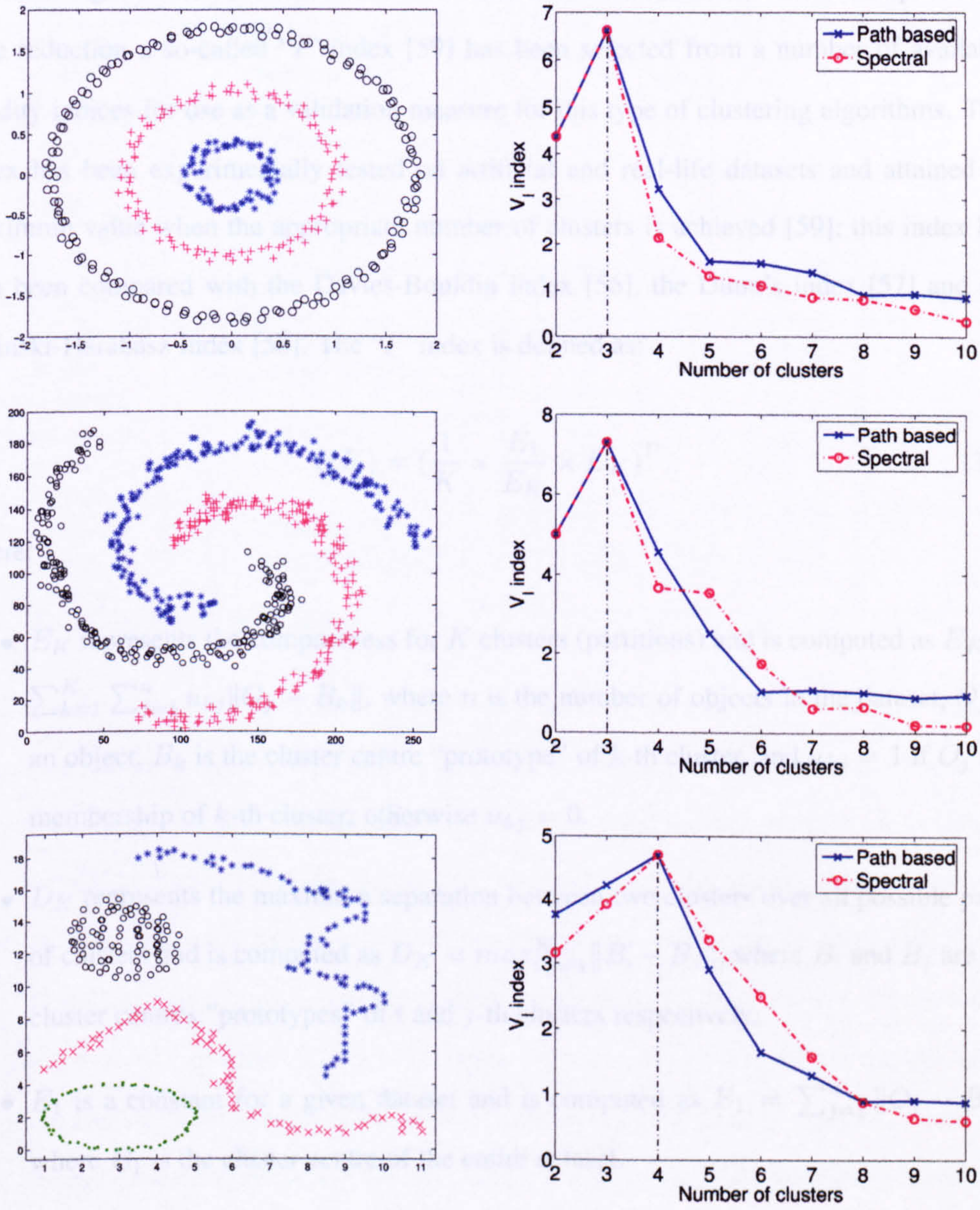
where K is the considered number of clusters.

For fuzzy clustering algorithms, a partition matrix $U(X) = [u_{ij}]_{K \times n}$ for the data is used to describe the membership degree of each data object to every cluster. In order to deal with fuzzy clustering algorithms, the proposed validity index is generalised as follows:

$$V_G = \frac{\sum_{i=1}^{K} Ie_i}{\sum_{i=1}^{K} \sum_{j=1}^{n} u_{ij} \times \max_{a,b \in T_{i-j}} \parallel x_a - x_b \parallel} \tag{3.2}$$

where x_a and x_b are two consecutive objects in T_{i-j} which is the minimum spanning tree of cluster i when object j belong to its membership.

Figure 3.3 shows the assessment of path based and spectral clustering results using V_I index. For the three-ring shaped dataset, the corresponding number of clusters K for which V_I validation index is maximised, is $K = 3$. The value of $K = 3$ represents the best fit number of clusters which actually coincides with the true number of clusters. Similarly, for the other datasets, the corresponding number of clusters K to the maximum value of V_I index is consistency referring to the number of clusters underlying the datasets.



(a) (b)

Figure 3.3: (a) Clustering results of different datasets having arbitrarily shaped clusters at the true number of clusters, and (b) the assessment using V_I validation index at different number of clusters.

In case of a structure bias to compact or spherical distributions

In case of clustering algorithms that have a structure bias to spherical distributions (e.g. K-means algorithm), V_I index can be used as a validation measure, but for computational time reduction a so-called "I" index [59] has been selected from a number of available validity indices for use as a validation measure for this type of clustering algorithms. This index has been experimentally tested on artificial and real-life datasets and attained its maximum value when the appropriate number of clusters is achieved [59]; this index has also been compared with the Davies-Bouldin index [56], the Dunn's index [57] and the Calinski-Harabasz index [58]. The "I" index is defined as:

$$I(K) = (\frac{1}{K} \times \frac{E_1}{E_K} \times D_K)^P \tag{3.3}$$

where:

- E_K represents the compactness for K clusters (partitions) and is computed as $E_K = \sum_{k=1}^{K} \sum_{j=1}^{n} u_{kj} \|O_j - B_k\|$, where n is the number of objects in the dataset, O_j is an object, B_k is the cluster centre "prototype" of k-th cluster, and $u_{kj} = 1$ if O_j is a membership of k-th cluster; otherwise $u_{kj} = 0$.

- D_K represents the maximum separation between two clusters over all possible pairs of clusters and is computed as $D_K = max_{i,j=1}^{K} \|B_i - B_j\|$, where B_i and B_j are the cluster centres "prototypes" of i and j-th clusters respectively.

- E_1 is a constant for a given dataset and is computed as $E_1 = \sum_{j=1}^{n} \|O_j - B_1\|$, where B_1 is the cluster centre of the entire dataset.

- P is a power used to control the contrast between different cluster configurations. In this study, $P = 2$ is used.

The value of K for which $I(K)$ is maximised is considered to be the correct number of clusters.

Table 3.1: Number of clusters estimated by different clustering algorithms using different validity indices and information criteria for six datasets.

Datasets	d	Clustering Algorithm	Validity index					Information criterion		
			DB	*Dunn*	*CH*	*I*	V_I	*AIC*	*MDL*	*CAIC*
Three-ring clusters (3 clusters)	2	Path based[31]	20	2	20	12	3	18	2	2
		Spectral [77] (unbiased)	18	2	18	9	3	19	14	14
Three-spirals arms (3 clusters)	2	Path based [31]	20	2	19	18	3	15	15	15
		Spectral [77] (unbiased)	12	2	16	8	3	16	16	16
Comm. data at 15 dB (4 clusters)	2	K-means [2]	4	4	4	4	4	4	4	4
		K-medoids [6]	4	4	4	4	4	4	4	4
Comm. data at 10 dB (4 clusters)	2	K-means [2]	3	2	5	4	4	4	4	4
		K-medoids [6]	3	2	4	4	4	4	4	4
Breast cancer data 1 (2 clusters)	9	K-means [2]	2	7	2	2	2	8	2	2
		K-medoids [6]	2	3	2	2	2	10	2	2
Breast cancer data 2 (2 clusters)	30	K-means [2]	2	6	2	2	2	4	2	2
		K-medoids [6]	2	14-19	2	2	2	7	2	2

where **d** is the number of dimensions

Table 3.1 shows the number of clusters estimated by each of the five validity indices (see Appendix A) and three information criteria from two clustering algorithms in each of the six datasets. The first type has arbitrarily shaped clusters (three-ring shaped clusters, and three spiral arms), while the second type has spherically distributed clusters (communication data, and breast cancer data). For three rings and spiral arms datasets, DB, Dunn, CH, "I" indices failed to estimate the number of clusters, as these indices are biased to spherically distributed clusters, while V_I index has successfully determined the true number of clusters in the underlying datasets. For communication data at 15 dB SNR, which has well-separated clusters, all cluster validity indices are able to provide the correct number of clusters. However, none of DB or Dunn indices are able to find the appropriate number of clusters for communication data at 10 dB SNR where the clusters are overlapped, while "I" and V_I indices provided the correct number of clusters, but CH index is partly successful. For breast cancer datasets, only Dunn index fails to indicate the correct number of clusters. For the purpose of comparison, the performance of the proposed V_I index is compared with information criteria [79, 80], namely, Akaike's information criterion (AIC), the consistent

Akaike's information criterion (CAIC), and minimum description length (MDL) criterion which formally coincides with the Bayesian inference criterion (BIC). As shown, for three rings and spiral arms datasets, AIC, MDL, CAIC information criteria methods failed to estimate the number of clusters, as these methods assume that the data are distributed according to a mixture of Gaussian distributions which favour spherical and hyperspherical distributed clusters. For communication data (which contains spherically distributed clusters), all information criteria methods are able to provide the exact number of clusters. For breast cancer data, only AIC failed to estimate the correct number of clusters as it tends to overestimate the number of clusters as reported in [81].

As demonstrated in Table 3.1, the proposed V_I index is found to be more consistent and reliable in indicating the correct number of clusters underlying the datasets, irrespective of the clustering algorithm and data distribution. However, it requires a high computational complexity compared with other validity indices, as it needs more time in computing the minimum spanning trees, where the computation cost for a minimum spanning tree [78] for cluster i is $O(m_i \, log \, n_i)$, with n_i denoting the number members belonging to cluster i and $m_i = n_i - 1$. These correspond to the number of vertex and edges in the spanning tree respectively. Therefore, for K clusters, the running time is $\sum_{i=1}^{K} O(m_i \, log \, n_i)$. Apart from V_I index, only "I" index is able to indicate the correct number of clusters for datasets that have spherical distributions, which is consistent with its validation principle. In addition, the complexity of "I" index is $O(Kn)$ which gives less computational time than V_I index. Therefore, for reducing the computation time, "I" index is used as a validation measure for clustering algorithms that are tendentious to spherical distributed clusters (e.g. K-means algorithm), and V_I index as a validation measure for clustering algorithms that have no such biases (e.g. path based clustering). This vindicates the choice of "I" and V_I indices as validation measures.

The assessment criterion is based on the ratio between the average value of the validity index values obtained from the r experiments and the maximum validation index value obtained from the used clustering algorithms; a higher value gives better robustness. The proposed validation measure assessment criterion of K partitions $Q_V(K)$ is defined as:

$$Q_V(K) = \frac{\frac{1}{r}\sum_{i=1}^{r} V_i}{V_{max}} \tag{3.4}$$

where $0 \leq Q_V(K) \leq 1$, V_i is the validation index value of run i for K partitions, and V_{max} is the maximum validation index value obtained from the used clustering algorithms. Figure 3.4 depicts the validation measure for communication data at SNR = 10 dB using the three clustering algorithms.

For each of the three clustering algorithms, Fig. 3.4 depicts the mean and the standard deviation of the r validity index values obtained from r experiments ($r = 100$) for every value of K. In these cases, the validity index possesses a clear maximum which corresponds to the number of clusters in the underlying the dataset. Moreover, the absence of a knee may be an indication that the dataset possesses no clustering structure. Additionally, the high variability of the validity index values at the same clustering scheme may impede the identification of the correct number of clusters as well as the robustness of the algorithm.

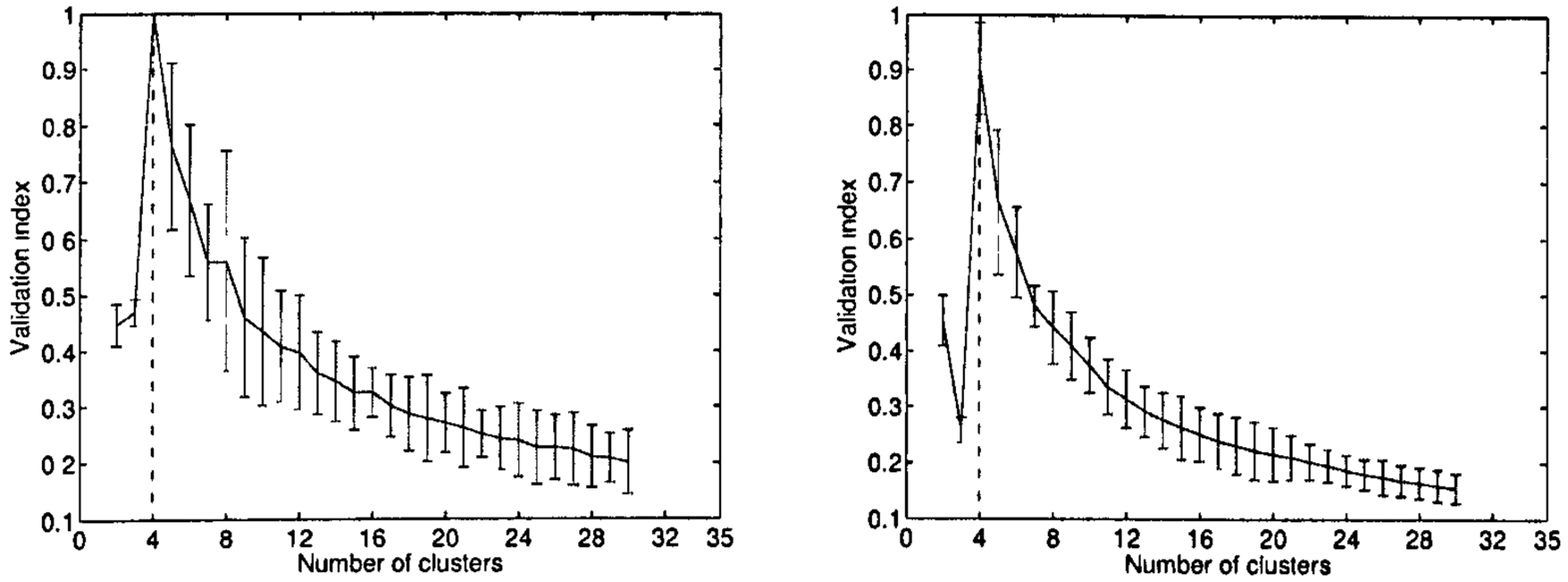
3.3.2 Repeatability measure

A novel stability measure is proposed, which is called a *repeatability* measure, to study the stability of a clustering algorithm. It is based on examining the common cluster memberships along a series of r experiments which are carried out for each set of parameter values. The procedure is described as follow:

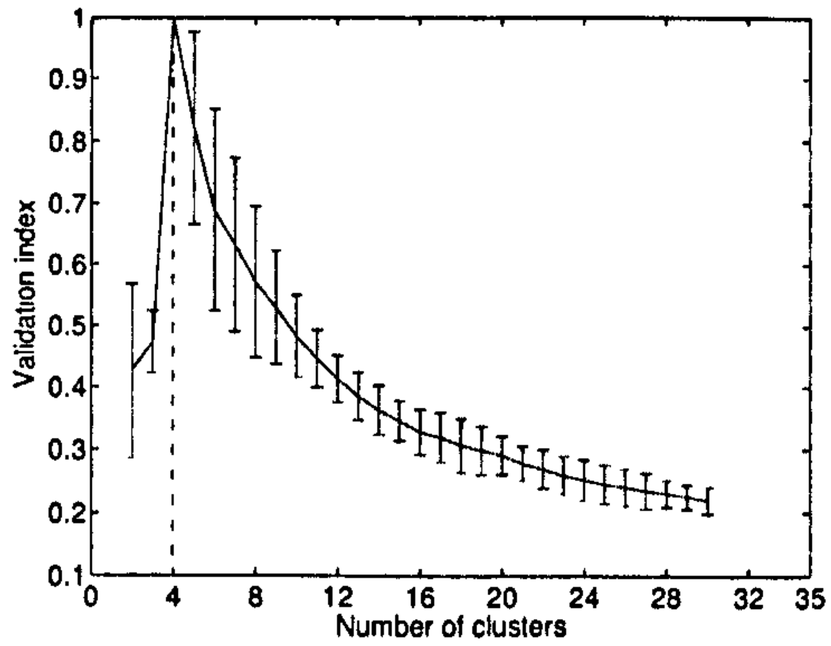
- Obtain r results from the same clustering scheme using either:

    1. The original dataset with different initialisation.

    2. Sub-samples from the datasets (r datasets).



(a) (b)



(c)

Figure 3.4: Validation measure of communications data with SNR = 10 dB, (a) K-means, (b) SOM, and (c) K-medoids.

3. r datasets (the original dataset plus extra samples) resulting in less separated clusters.

- Select a reference result out of the r results obtained above by using a reliable validity index to get the best one.

- Determine the number of runs R that achieve the common cluster memberships with respect to reference clusters (η) for each cluster of the reference run, where η is the ratio between the number of common cluster memberships and the size of the reference cluster. In this study, $\eta = 0.8$ is used.

- For each cluster of the reference run:

  1. Calculate the gain per run (N/M) that equals the ratio between the number of common members (N) obtained by the R runs in the comparison cluster and the number of members (M) (size of reference run cluster) in the reference cluster.

  2. Calculate the weighted gain, $(N/M) * R^m$, where m is used to control the contrast between the number of runs R that achieve the common cluster memberships in the reference cluster. In this study, $m = 2$ is used.

The proposed *repeatability* measure is presented in algorithm 1.

Figure 3.5(a) and 3.5(b) depict the repeatability measure of SOM algorithm on 10 dB SNR communication data at $r = 100$ with different initialisations and $\eta = 0.8$. The two sub-figures contain three subplots each, these subplots show the difference in magnitude between the size of reference run clusters (M) and the comparison clusters (N) (topmost plot), the number of runs R that achieve the common cluster membership for each cluster in the reference run (middle plot), the gain per run (N/M) and the weighted gain per run $(N/M) * R^m$ (bottom plot).

Algorithm 1 Repeatability measure procedure

MV is the run member vector of size equal to the dataset length and contains clusters memberships.

For each cluster of the reference run $C_i, i = 1, \ldots, K_{ref}$

 # Find the cluster membership MC_i *of* C_i *and its size* M_i

$MC_i = (MV_{ref} == C_i)$

$M_i = sum(MV_{ref} == C_i)$.

For each run x *of the* r *runs,* $x = 1, \ldots, r$

For each cluster C_j *of run* x, $j = 1, \ldots, K_j$

 # Find the common cluster membership between C_i & C_j and its size M_{ij}.

$MC_{ij} = MC_i \cap MC_j$

$M_{ij} = sum(MC_{ij} == 1)$

End

 # Find the best cluster C_j *that achieve* $\frac{M_{ij}}{M_i} \geq \eta$ *with* C_i

$M_{ibest} = \max_{j=1 \ldots K_j} M_{ij}$

if $M_{ibest} \geq \eta$ *then*

$MC_i = MC_{ibest}$

$R = R + 1$

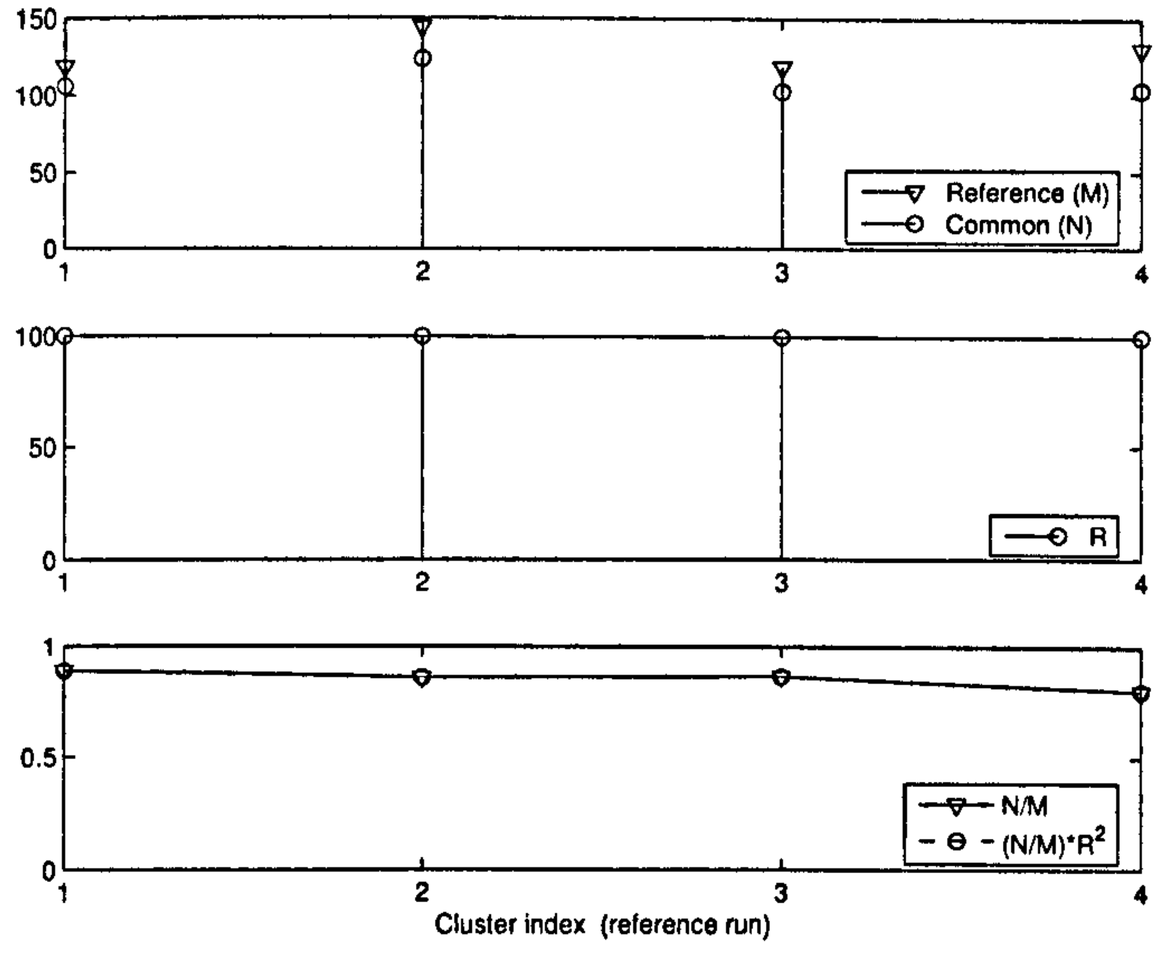
End

End

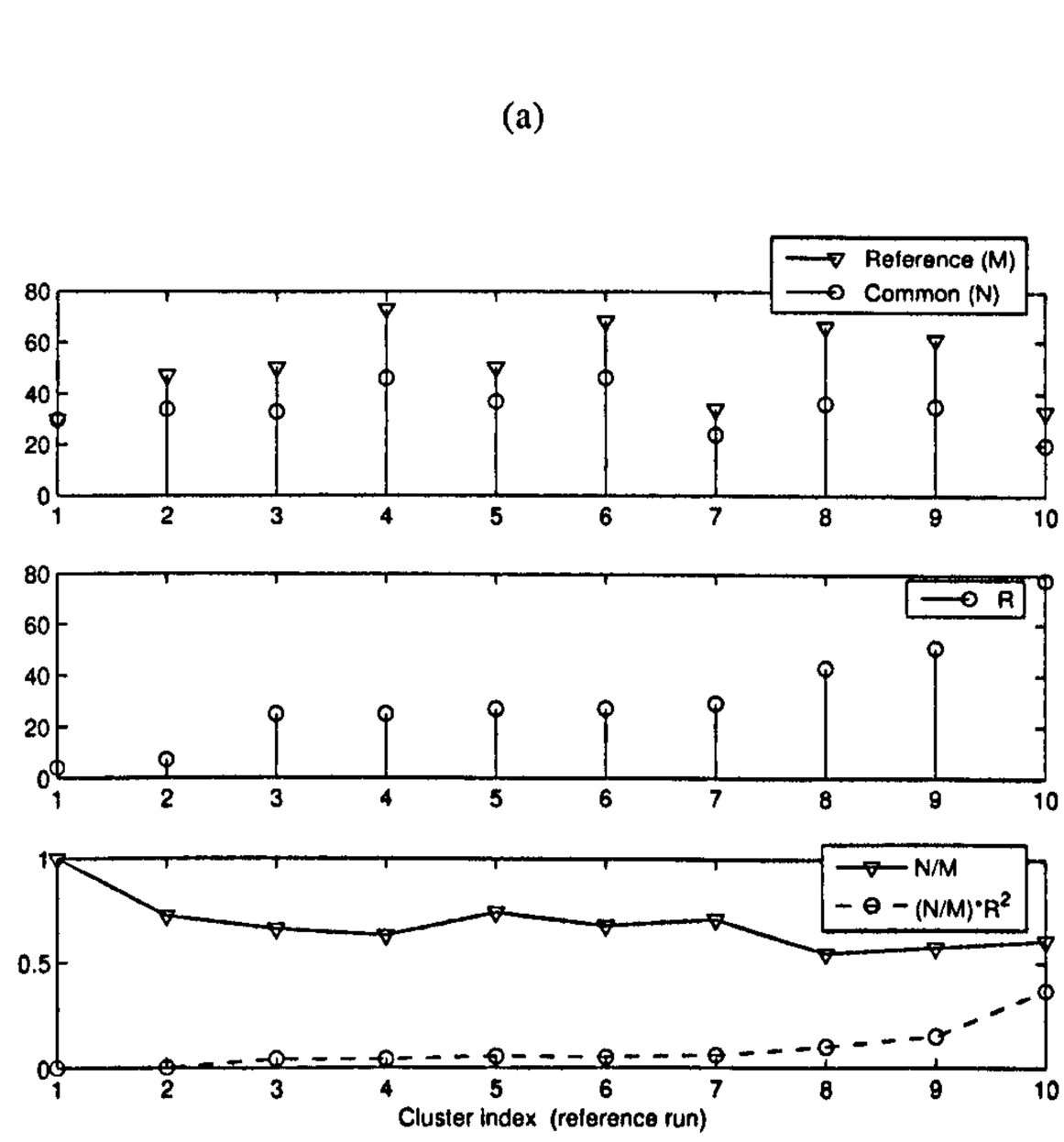
The proposed assessing criterion is based on the expected value of the weighted gain; a higher value gives better repeatability. Thus, a good clustering algorithm is the one that can achieve higher values for many possible sets of parameter values. The proposed repeatability measure assessment criterion of K partitions $Q_R(K)$ is defined as:

$$Q_R(K) = E((N/M) * R^m) \tag{3.5}$$

where $0 \leq Q_R(K) \leq 1$.



(a)



(b)

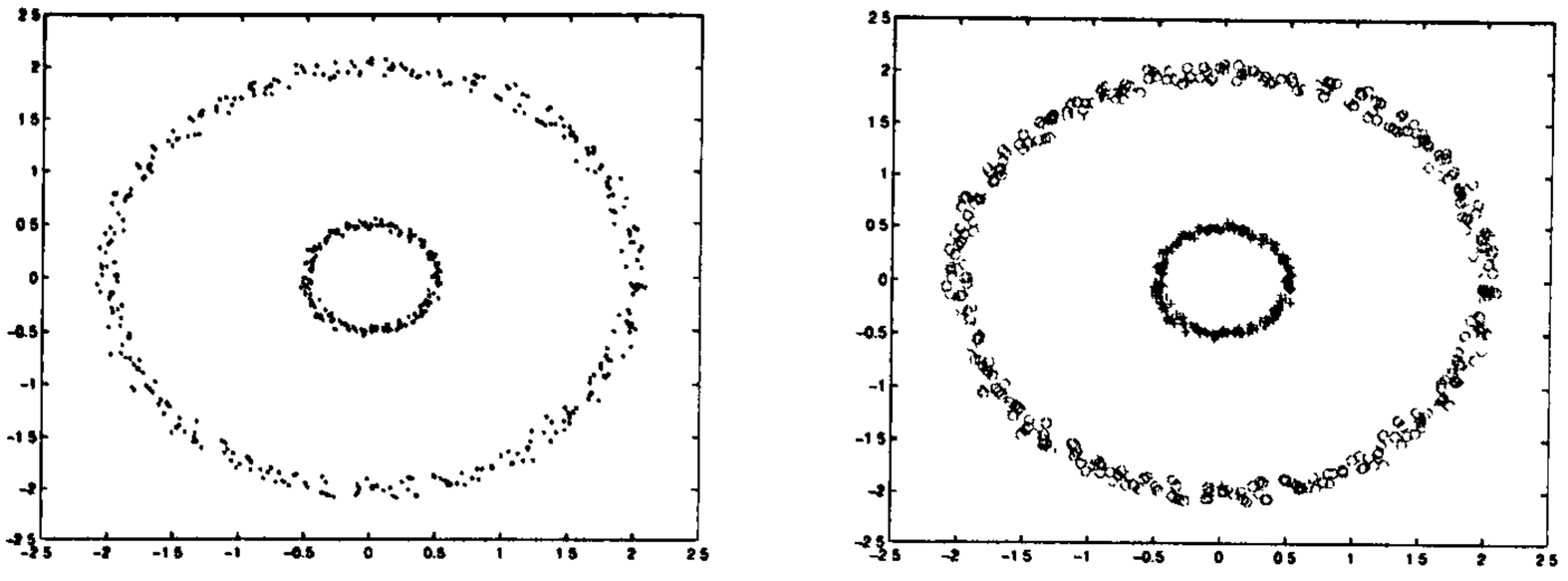
Figure 3.5: Repeatability measure of the SOM algorithm on communication data with SNR = 10 dB at: (a) $K = 4$, and (b) $K = 10$.

Table 3.2: Estimated number of clusters by different methods.

dataset	Repeatability	Gap statistic [66]	Stability method [68]	Bootstrapping [67]	True clusters
QAM-4 15 dB (K-means)	4	4	4	4	4
QAM-4 10 dB (K-means)	4	2	4	2	4
Breast cancer 1 (K-means)	2	2	2	2	2
Breast cancer 2 (K-means)	2	8	2	2	2
2 Ring-shaped (Path based)	2	1	2	2	2

Competing repeatability methods for estimating the number of clusters

It should be noted that there are other choices for a repeatability measure like Gap statistic [66], Stability-based validation method [68], and cluster validity by Bootstrapping [67]. In the following these alternative measures are compared using different datasets and different clustering algorithms. As shown in Table 3.2, all methods are able to estimate the correct number of clusters for QAM-4 (SNR = 15 dB) and breast cancer dataset 1. The Gap statistic fails in finding the correct number of clusters for QAM-4 (SNR = 10 dB) which indicates its shortfalls for noisy datasets, in addition to the drawbacks mentioned in the introduction section. Also, it fails with the breast cancer dataset 2 in which the effect of the comparison with random data may not scale well for higher dimensional datasets. For bootstrapping, experiments have highlighted the sensitivity of this method for noisy datasets which is clearly observed in QAM-4 (SNR = 10 dB). In order to determine whether the proposed repeatability measure contains a structure bias or not, the path based clustering algorithm [31] is applied on a dataset contain two ring-shaped clusters as shown in Fig. 3.6. As shown in Table 3.2, the proposed repeatability measure can infer the correct number of clusters as stability-based validation method and bootstrapping validation method compared against Gap statistic that infers $K = 1$ as the correct number of clusters, because it directly incorporates the assumption of spherically distributed datasets. Overall, both the proposed repeatability measure and the stability-based validation method can infer the correct num-



(a) (b)

Figure 3.6: (a) Dataset containing two ring-shaped clusters, and (b) path based clustering algorithm performance at $K = 2$.

ber of clusters in each of the five datasets used, as shown in Table 3.2, while the Gap statistics and the bootstrapping methods are only partially successful.

3.3.3 Clustering performance measures (CPMs)

Two important aspects are considered to determine the reliability of a clustering algorithm; a repeatability measure is used for studying the stability of a clustering algorithm in terms of the common clusters memberships of several clustering results, and validation measure is used for measuring the robustness, quality, of the clustering results.

3.3.3.1 Clustering performance measure 1 (CPM_1)

This proposed assessment criterion is based on the weights of the tested partitions. Let $Q_R(K)$ and $Q_V(K)$ be the repeatability and validation values of the K partitions. Now it is worth referring to the Fig. 3.7, which plots Q_R versus Q_V. Each experimental result will be represented by one point on this plot. It should be remarked that the ideal performance will be achieved by a point with $Q_R = 1$ and $Q_V = 1$ (i.e. the top right-hand corner of the plot). Any point further away from this corner represents lower performance. This can be modelled by a weight corresponding to the reciprocal of the distance between the point and the ideal position of $(Q_R, Q_V) = (1, 1)$. However such a ratio may produce

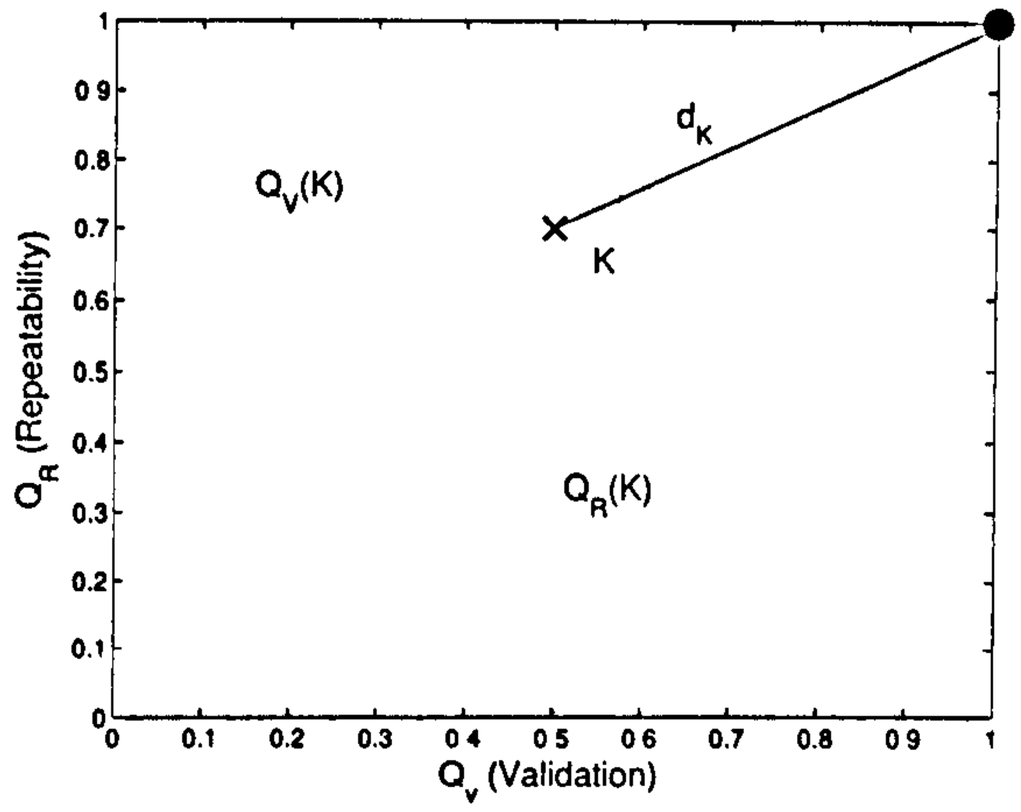


Figure 3.7: Clustering performance measure 1 - basis.

a singularity when the distance is zero. To avoid such a singularity and yet to capture the previously described trend, the weight of the K partitions is defined as $\frac{1}{1+d_K}$, where $d_K = \sqrt{(1 - Q_V(K))^2 + (1 - Q_R(K))^2}$, $d_K \geq 0$. The clustering performance measure, CPM_1, is defined as:

$$CPM_1 = \sum_{K=2}^{N_k} \frac{1}{1 + d_K} \tag{3.6}$$

where N_k is the number of tested partitions that are considered in the evaluation procedure.

This takes into account all the points (partitions) and ensures that larger values of CPM_1 represents better clustering performance.

3.3.3.2 Clustering performance measure 2 (CPM_2)

This clustering performance measure is based on the contribution of the repeatability measure and validation measure on the evaluation procedure. The assessment criterion, CPM_2, is defined as:

$$CPM_2 = E(Q_R) + E(Q_V) - E(Q_R)E(Q_V) \tag{3.7}$$

where:

$CPM_2 \geq max(E(Q_R), E(Q_V))$, $E(Q_R) = \frac{1}{N_k} \sum_{K=2}^{N_k} Q_R(K)$ is the expected value of

the weighted gain (*repeatability measure*), $E(Q_V) = \frac{1}{N_k}\sum_{K=2}^{N_k} Q_V(K)$ is the expected value of the corresponding validity index values (*validation measure*), where N_k is the number of tested partitions that are considered in the evaluation procedure, $Q_R(K)$ is the repeatability value of K partitions, and $Q_V(K)$ is the validation value of K partitions.

From Eqs. 3.6 and 3.7, a higher value of the overall performance gives more reliability to the clustering algorithm. Moreover, these equations can be used to compare clustering algorithms from the two important points of view - the repeatability and robustness.

For the trivial case $K = 1$, all clustering solutions will be the same, so there is no need for any computation in this case.

In this study, five relatively common clustering algorithms are evaluated by the proposed CPMs. Three of the clustering algorithms have a structure bias to spherical distributions, namely, K-means [1, 2], K-medoids [6], Self Organising Map (SOM) [29], The other two algorithms are spectral clustering, and path based clustering which have no structure biases [30, 31, 77].

3.4 Experimental Results

The degree of stability of a clustering algorithm is investigated using well-separated datasets and datasets containing outliers as well as overlapped clusters.

3.4.1 For algorithms that have a structure bias to spherical distributions

3.4.1.1 Considering the effect of initial conditions

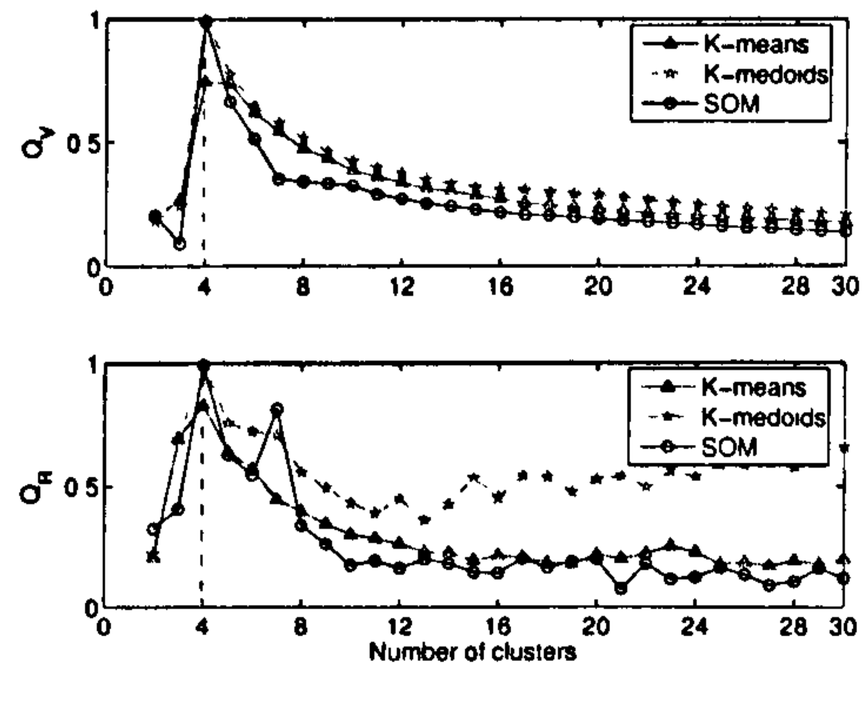
In this sub-section, the effect of initial conditions on the cluster memberships of clustering algorithms that have initialisation dependency will be examined.

Communications data

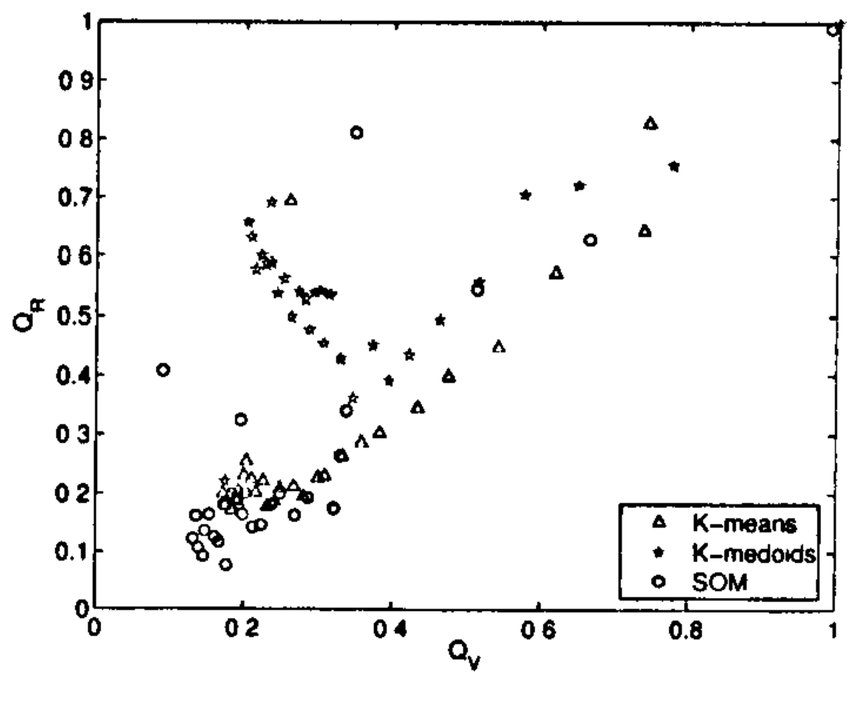
A series of experiments ($r = 100$) for every K ($K = 2, \ldots, 30$) with $\eta = 0.8$, using different initialisation each time, were carried out on different SNR values, SNR = 15 dB and SNR = 10 dB using three clustering algorithms. Thus a total of 17,400 experiments were carried out. The validation measure and repeatability measure are used for the evaluation procedure. In these experiments, "I" validity index is used as a validation measure.

Figures 3.8 and 3.9 contain two plots each, part (a) contains two subplots: the top subplot shows the mean validation index (of 100 runs) of three different clustering algorithms versus the number of clusters that are considered above, where the number of clusters corresponding to the "knee" point are expected to be the true number of clusters. Furthermore, higher validation values indicate better robustness. The bottom subplot shows the repeatability of the three clustering algorithms over the same number of clusters considered above. Clearly, the resistance of the clustering algorithm against the noise and outliers is identified by detecting the cluster members that remain together regardless of the initial conditions. Furthermore, the higher value represents more repeatability at the given number of clusters. As shown, the number of clusters corresponding to the "knee" points of the repeatability for the three clustering algorithms are the true number of clusters that underlying the dataset, i.e. the estimated number of clusters is corresponding to the higher repeatability value. Part (b) shows the relation between the validation index values and their repeatability, where each point has two coordinates; the repeatability value $Q_R(K)$ of K partitions and its corresponding validation value $Q_V(K)$; a better clustering algorithm is the one that achieves higher validation as well as repeatability.

Tables 3.3 and 3.4 contain four values per clustering algorithm. The first is the repeatability value which is the expected value of the weighted gain (*repeatability measure*) over all the number of tested partitions that are considered in the evaluation procedure. The second is the validation value which is the the expected value of the validity index values (*validation measure*) over all the number of tested partitions that are considered in the evaluation procedure. Finally, the third and the fourth are the corresponding CPM values.



(a)

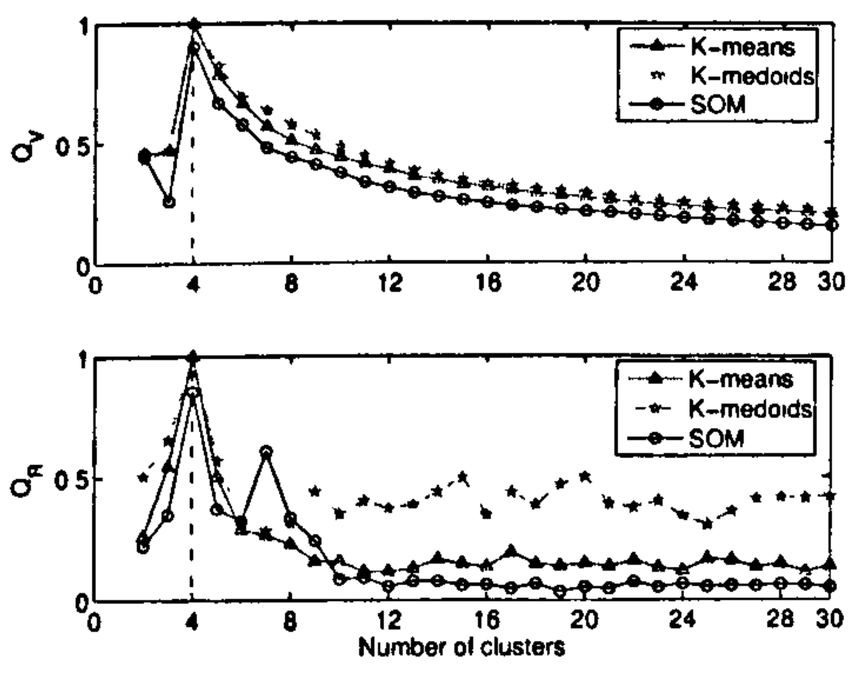


(b)

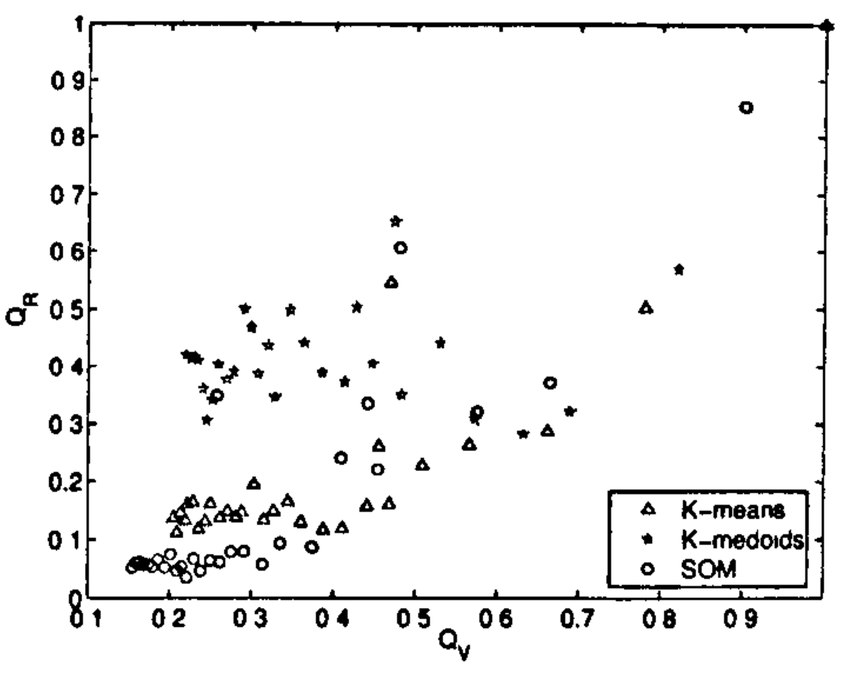
Figure 3.8: (a) Testing rule procedure on different clustering algorithms for QAM-4 at SNR = 15 dB, and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.3: Results of different clustering algorithms on QAM-4 at SNR = 15 dB communication dataset.

	SOM	K-means	K-medoids
$< Repeatability >$	0.26	0.30	0.55
$< Validation >$ ("I" index)	0.26	0.31	0.36
CPM_1	14.53	14.85	16.55
CPM_2	0.45	0.52	0.71



(a)



(b)

Figure 3.9: (a) Testing rule procedure on different clustering algorithms for QAM-4 at SNR = 10 dB, and (b) relationship between validation (Q_V) and repeatability (Q_R).

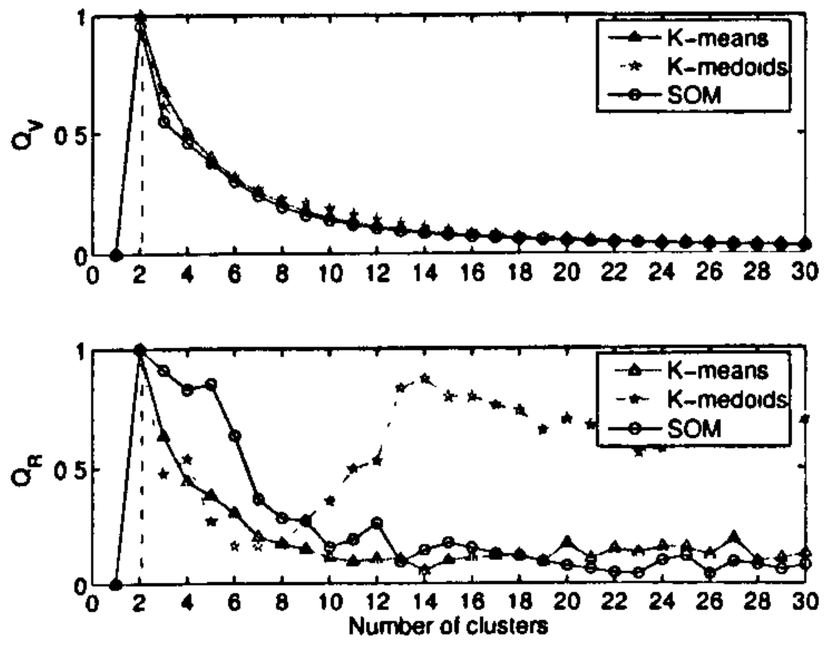
Table 3.4: Results of different clustering algorithms on QAM-4 at SNR = 10 dB communication dataset.

	SOM	K-means	K-medoids
$< Repeatability >$	0.16	0.21	0.43
$< Validation >$ ("I" index)	0.31	0.38	0.40
CPM_1	14.14	14.80	16.08
CPM_2	0.42	0.51	0.66

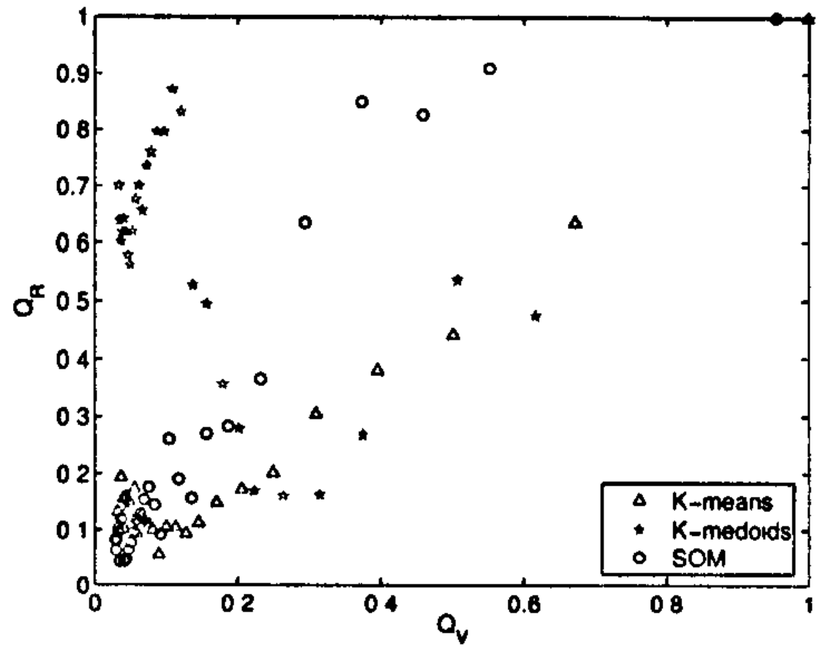
As shown in Table 3.8, the SOM algorithm achieves lower validation as well as repeatability than K-means and K-medoids clustering algorithms. As the SNR value decreases, the SOM algorithm loses its level of validation as well as repeatability as shown in Fig. 3.9 and Table 3.4. Additionally, the repeatability is not affected by decreasing the SNR value and K-medoids appears to offer higher repeatability. Finally, one can deduce that K-medoids algorithm possesses higher robustness as well as repeatability and K-means algorithm performs well for noisy dataset compared to SOM algorithm in addition to its simple design.

Breast cancer data

A series of experiments ($r = 100$), for every K ($K = 2, \ldots, 30$), were carried out for each of the two breast cancer datasets. Thus a total of 17,400 experiments were carried out. Figure 3.10 and 3.11 show the testing rule procedure results of breast cancer dataset 1 and 2. Although, the overall performance of SOM algorithm is better than K-means algorithm for breast cancer dataset 1, it gives the lower performance for breast cancer dataset 2, This is because the lower compactness of clusters on breast cancer dataset 2 compared against the breast cancer dataset 1 which coincides with the conclusion obtained from the experiments that have been done on the communication datasets. From Table 3.5 and 3.6, one can deduce that the overall performance of K-medoids algorithm is better than SOM algorithm and K-means algorithm.



(a)

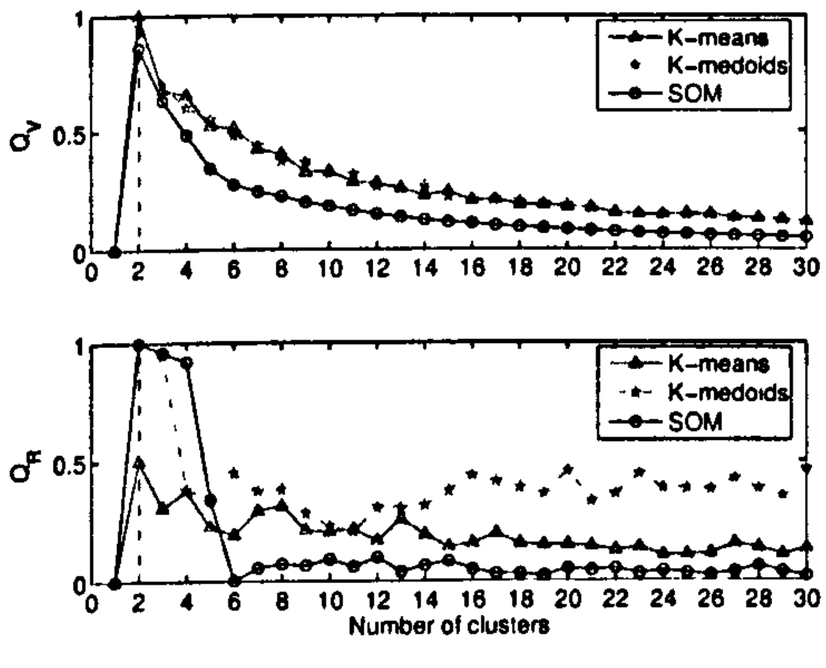


(b)

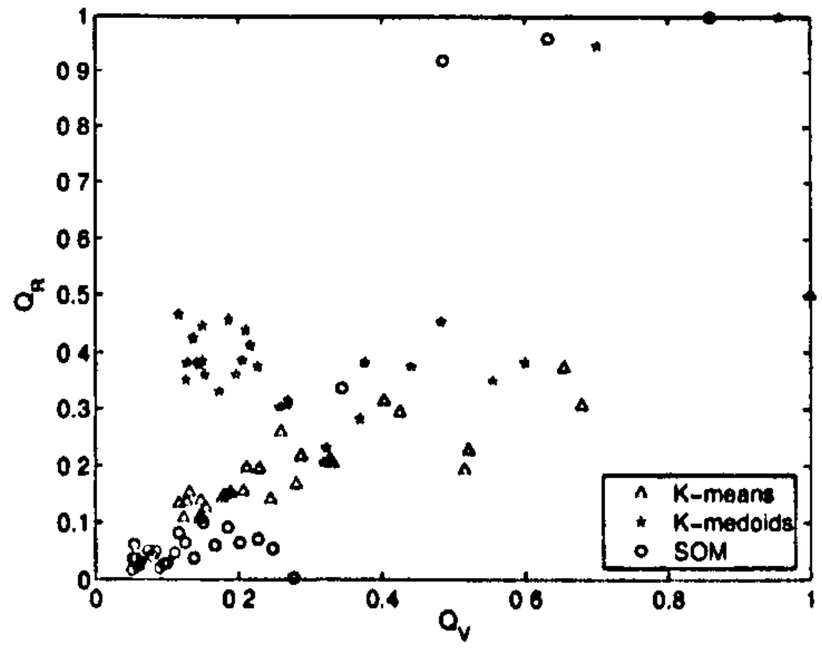
Figure 3.10: (a) Testing rule procedure on different clustering algorithms for breast cancer dataset 1, and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.5: Results of different clustering algorithms on breast cancer dataset 1.

	SOM	K-means	K-medoids
$< Repeatability >$	0.26	0.20	0.58
$< Validation >$ ("I" index)	0.15	0.17	0.18
CPM_1	14.06	13.88	15.16
CPM_2	0.37	0.34	0.65



(a)



(b)

Figure 3.11: (a) Testing rule procedure on different clustering algorithms for breast cancer dataset 2, and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.6: Results of different clustering algorithms on breast cancer dataset 2.

	SOM	K-means	K-medoids
$< Repeatability >$	0.15	0.19	0.41
$< Validation >$ ("I" index)	0.18	0.29	0.30
CPM_1	13.66	14.08	15.37
CPM_2	0.30	0.43	0.58

3.4.1.2 Ignoring the effect of initial conditions

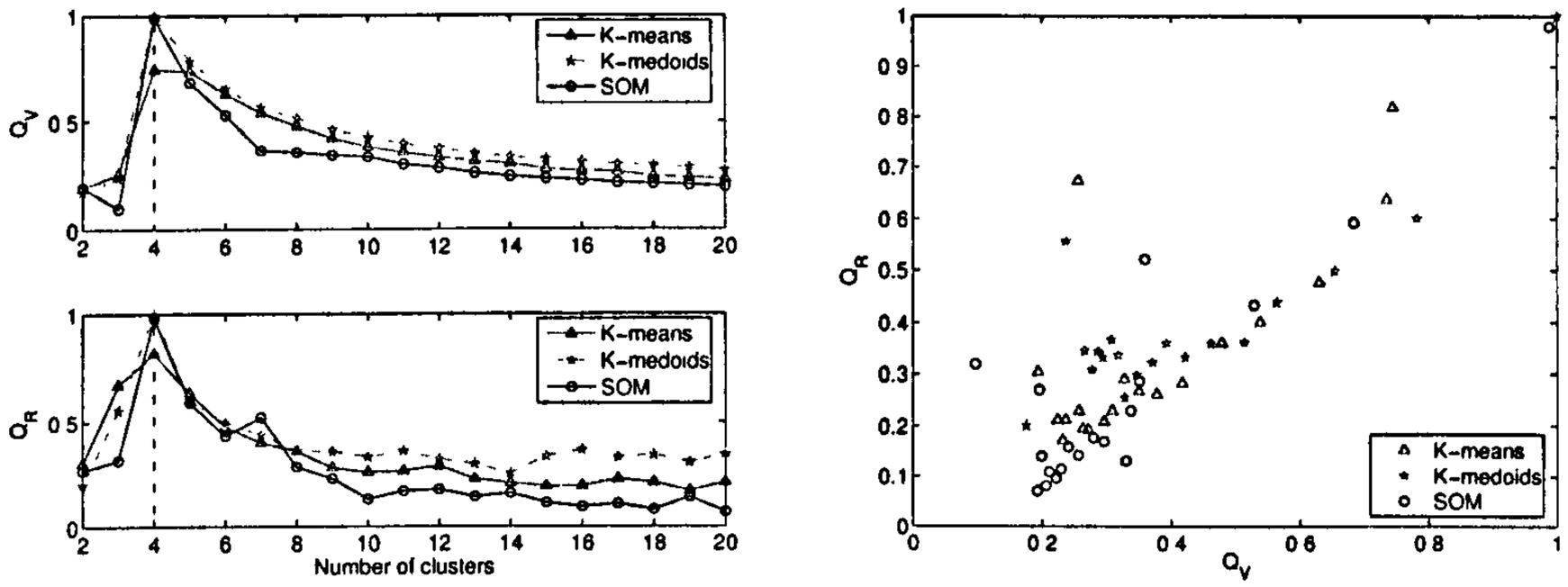
In order to confirm the conclusion obtained above, similar experiments were carried out on:

- Sub-samples from the datasets (r datasets).
- r datasets (the original dataset plus extra samples) resulting in less separated clusters.

Sub-sampling

The idea is based on randomly selecting samples from the dataset with a sampling ratio S_R (a proportion of data points sampled) where $S_R < 1$. In this study, $S_R = 0.8$ is used. Therefore, one can get r datasets, each of which is sampled from the original dataset. Then, a series of experiments are carried out for every sub-sampled dataset ($r = 1, \ldots, 100$) with K ($K = 2, \ldots, 20$), i.e. one can get r clustering solutions for every K. Therefore, a total of 22,800 experiments were carried out. However, the proposed CPMs cannot be applied, because there are missing samples for each clustering solution. To overcome such a problem, each missing sample is assigned to the most appropriate cluster that is consistent with the clustering algorithm. In the case of K-means algorithm, the missing sample is assigned to the cluster with the closest centre. It should be remarked that the value of S_R should be greater than 0.5; otherwise not all clusters may appeared in the sub-samples.

As shown in Figures 3.12, 3.13, 3.14, and 3.15, and Tables 3.7, 3.8, 3.9, and 3.10, the clustering performance for the three clustering algorithms is consistent with the performance obtained in the experiments on the original datasets without sub-sampling.

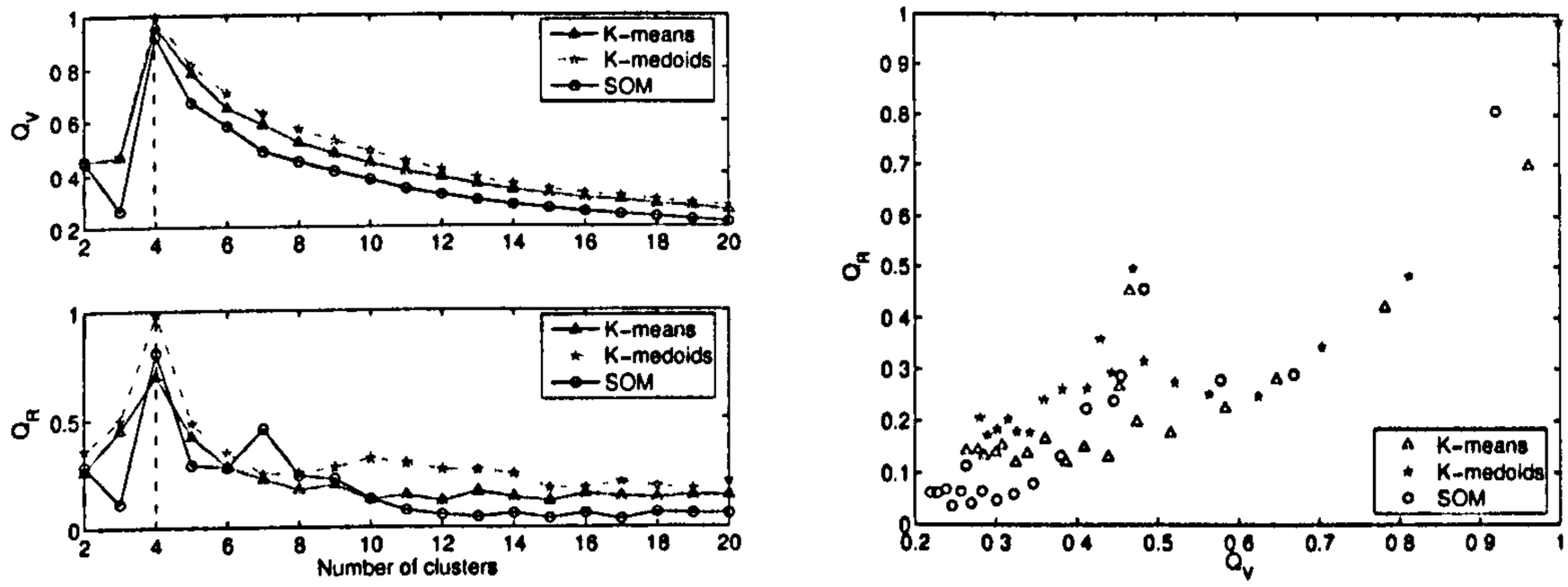


(a) (b)

Figure 3.12: (a) Testing rule procedure on different clustering algorithms for QAM-4 at SNR = 15 dB (sub-sampled datasets), and(b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.7: Results of different clustering algorithms on sub-sampled QAM-4 at SNR = 15 dB communication dataset.

	SOM	K-means	K-medoids
$< Repeatability >$	0.26	0.34	0.40
$< Validation >$ ("I" index)	0.33	0.38	0.42
CPM_1	9.83	10.08	10.63
CPM_2	0.50	0.59	0.65

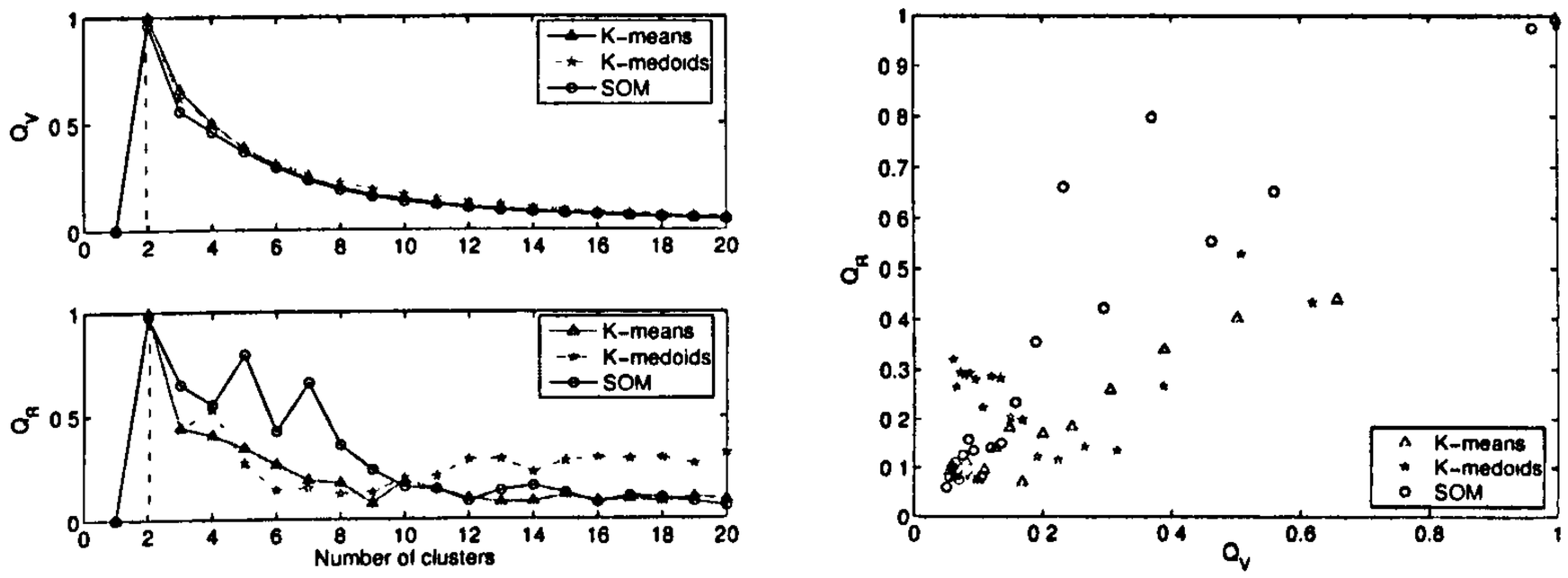


(a) (b)

Figure 3.13: (a) Testing rule procedure on different clustering algorithms for QAM-4 at SNR = 10 dB (sub-sampled datasets), and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.8: Results of different clustering algorithms on sub-sampled QAM-4 at SNR = 10 dB communication dataset.

	SOM	K-means	K-medoids
$< Repeatability >$	0.18	0.22	0.31
$< Validation >$ ("I" index)	0.38	0.45	0.48
CPM_1	9.56	9.85	10.43
CPM_2	0.49	0.57	0.64

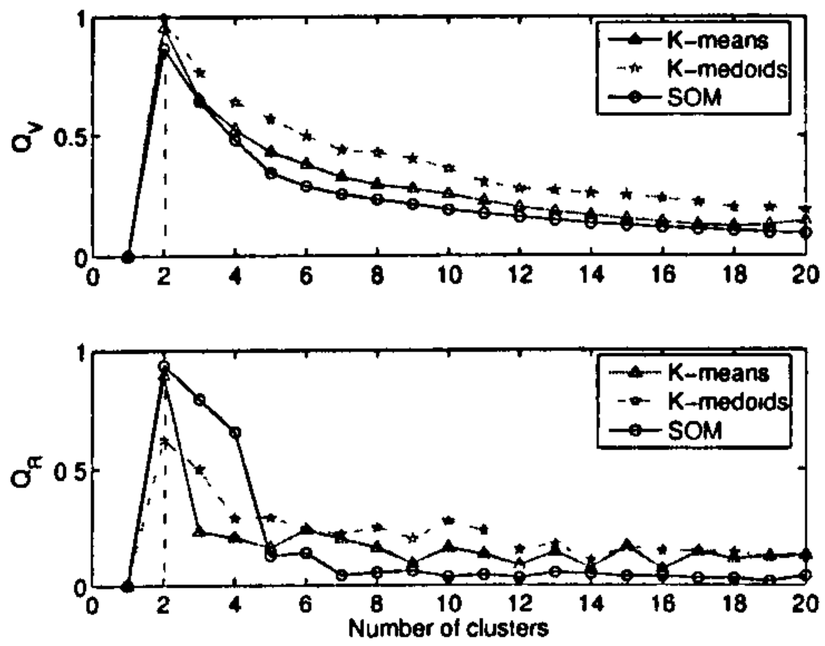


(a) (b)

Figure 3.14: (a) Testing rule procedure on different clustering algorithms for sub-sampled breast cancer dataset 1, and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.9: Results of different clustering algorithms on sub-sampled breast cancer dataset 1.

	SOM	K-means	K-medoids
$< Repeatability >$	0.31	0.21	0.30
$< Validation >$ ("I" index)	0.22	0.23	0.25
CPM_1	9.62	9.39	9.64
CPM_2	0.46	0.39	0.48



(a) (b)

Figure 3.15: (a) Testing rule procedure on different clustering algorithms for sub-sampled breast cancer dataset 2, and (b) relationship between validation (Q_V) and repeatability (Q_R).

Table 3.10: Results of different clustering algorithms on sub-sampled breast cancer dataset 2.

	SOM	K-means	K-medoids
$< Repeatability >$	0.17	0.19	0.23
$< Validation >$ ("I" index)	0.25	0.29	0.39
CPM_1	9.27	9.36	9.70
CPM_2	0.37	0.43	0.53

Creating a dataset of less separated clusters

Randomly, 20% extra samples are added to the original datasets in the range of the observed values for each feature; effectively this constitutes adding noise. Therefore, one can get r new datasets. Then, a series of experiments are carried out for every dataset ($r = 1, \ldots, 100$) with K ($K = 2, \ldots, 20$), i.e. one can get r clustering solutions for every K. A total of 28,500 experiments has been carried out for this part of investigation. The proposed CPMs are applied after removing the extra samples from the clustering solutions. As shown in Tables 3.11, 3.12, 3.13, and 3.14, both CPM_1 and CPM_2 point consistently to the superiority of K-medoids over K-means and SOM. Moreover, the effect of the extra samples can be observed on the overall clustering performance of SOM algorithm due to the lower compactness of the clusters on the datasets which coincides with the conclusion obtained from the experiments that have been done on the original datasets. It should be

remarked that the number of extra samples constitutes no more 25% of the original dataset; otherwise, the distribution of the clusters will be lost as the dataset will not possesses clustering structures.

Table 3.11: Results of different clustering algorithms on QAM-4 at SNR = 15 dB communication dataset after adding extra samples.

	SOM	K-means	K-medoids
$< Repeatability >$	0.31	0.36	0.48
$< Validation >$ ("I" index)	0.32	0.40	0.43
CPM_1	9.94	10.51	11.06
CPM_2	0.53	0.61	0.70

Table 3.12: Results of different clustering algorithms on QAM-4 at SNR = 10 dB communication dataset after adding extra samples.

	SOM	K-means	K-medoids
$< Repeatability >$	0.21	0.28	0.37
$< Validation >$ ("I" index)	0.37	0.44	0.45
CPM_1	9.65	10.26	10.63
CPM_2	0.50	0.59	0.65

Table 3.13: Results of different clustering algorithms on breast cancer dataset 1 after adding extra samples.

	SOM	K-means	K-medoids
$< Repeatability >$	0.24	0.25	0.46
$< Validation >$ ("I" index)	0.21	0.23	0.24
CPM_1	9.38	9.62	10.17
CPM_2	0.41	0.43	0.59

Table 3.14: Results of different clustering algorithms on breast cancer dataset 2 after adding extra samples.

	SOM	K-means	K-medoids
$< Repeatability >$	0.16	0.21	0.27
$< Validation >$ ("I" index)	0.25	0.29	0.38
CPM_1	9.24	9.44	10.01
CPM_2	0.37	0.44	0.55

3.4.2 For algorithms that have no structure bias

To evaluate the clustering algorithms that have no structure biases using the proposed CPMs, similar experiments were carried out using the sub-sampling criterion as demonstrated in Section 3.4.1.2. A total of 7,600 experiments were carried out. In these experiments, V_I validity index is now used as a validation measure. Furthermore, these experiments are carried out on two synthetic datasets containing arbitrarily shaped clusters. Tables 3.15 and 3.16 show the testing rule procedure results. As shown, the path based clustering algorithm achieves low level of repeatability compared with spectral clustering algorithm. However, it consistently offers higher validation than spectral clustering algorithm. For spiral data, to some extent, the decrease in the repeatability of path based clustering is met by an increase in the validation of spectral clustering with the same amount which leads to comparable performance of both algorithms as demonstrated in Table 3.16. This vindicates the necessity for correlating the repeatability measure and validation measure to obtain a reliable and fair evaluation.

Table 3.15: Results of different clustering algorithms on dataset containing three-ring shaped clusters.

	Path based clustering	Spectral clustering
$< Repeatability >$	0.23	0.25
$< Validation >$ ("V_I" index)	0.40	0.31
CPM_1	9.88	9.62
CPM_2	0.54	0.48

Table 3.16: Results of different clustering algorithms on dataset containing three spirals arms.

	Path based clustering	Spectral clustering
$< Repeatability >$	0.37	0.41
$< Validation >$ ("V_I" index)	0.29	0.25
CPM_1	10.0	9.90
CPM_2	0.56	0.55

3.5 Summary

In this chapter, a new validity index V_I and a *repeatability measure* have been proposed. The V_I index has been used to assess results of clustering algorithms that are able to cluster the datasets with arbitrarily shaped clusters, while the *repeatability measure* has been used to study the stability of clustering results. Comparisons with other methods indicate the applicability of the proposed validity index, V_I index, and *repeatability measure* in estimating the number of clusters correctly. In addition, new measures (CPMs) for the robustness and reliability of clustering algorithms have been proposed. These CPMs are used to evaluate clustering algorithms that have a structure bias to certain types of data distribution as well as those that have no such biases. As demonstrated, the effect of initial and random guesses on the clustering algorithms that have initialisation dependency is tested and evaluated using different types of real-world data and synthetic data which contain well-separated datasets, as well as overlapped datasets. These measures have been applied on sub-sampled datasets and datasets with less separated clusters. Therefore, one can use the proposed CPMs to evaluate the clustering algorithms that have a unique solution for a given set of parameter values with no initialisation dependency.

Chapter 4

Radius Based Clustering Algorithm (RACAL)

4.1 Introduction

CLUSTERING algorithms are becoming an ever more common tool for use in different real-life applications. Many different clustering algorithms exist, and these can be used to find different numbers of cluster centres depending on the requirements of the particular problem. In well bounded and understood datasets, it is relatively easy to determine how many different clusters are required. Once this number is known, then it is relatively easy to perform the clustering and interpret the results. However, when the data is either noisy or not easily separable (which is often the case with many different real-world datasets), it can be much more difficult to determine with confidence the true number of centres for the clustering algorithm.

In this chapter, a new clustering algorithm, which is a RAdius based Clustering ALgorithm (*RACAL*), is proposed. The proposed algorithm uses a distance based principle to map the distributions of the data assuming that clusters are determined by a distance parameter, without having to specify the number of clusters. The proposed clustering algorithm is enhanced by a reliable validity index to choose the best clustering results for a given input parameter. Additionally, an adaptive partial supervision strategy is proposed

for use in conjunction with *RACAL* to make it act as a classifier. Furthermore, a parallel algorithm for *RACAL* is proposed to reduce the clustering time of large datasets.

4.2 The RACAL Clustering Algorithm

The basic idea of the proposed clustering algorithm is to find the proper prototypes, cluster centres, that can map the distributions on datasets at a given input parameter value without neglecting the sparsely populated areas as in density based approaches and it does not require the knowledge of the true number of clusters. The proposed algorithm uses a distance based principle, which fundamentally differs from density based methods in the way that the algorithm determines what constitutes a cluster. Simply expressed, the proposed algorithm defines a normalised distance parameter, δ_o ($0 \leq \delta_o \leq 1$), which acts as the determinant of the cluster. From a given object which is characterised by d features, any other objects that fall within δ_o are regarded as belonging to the same cluster, i.e., have similar features. The control of the cluster size is achieved through the value of δ_o parameter. Small values will lead to a high number of small and tight clusters, while large values of δ_o will create a smaller number of larger clusters. Extremely large values will cause only one cluster to be formed.

Clustering procedure

Clustering is a process of grouping objects into clusters in such a way that each object within a cluster is close or similar to one another, but dissimilar from the objects in other clusters. This section presents the proposed *RACAL* algorithm to cluster a dataset. Let $O = \{O_i | i = 1, \ldots, n\}$ be a set of n objects, where each object, $O_i \in R^d$, is characterised by d features. As a first step, obtain the relational matrix "normalised distance matrix" $R = [r_{ij}]$, where r_{ij} indicates the relative distance between O_i and O_j, and satisfies the following conditions:

$$r_{ij} = r_{ji}, r_{ii} = 0, \text{and } r_{ij} \in [0, 1]$$

Then, search for the proper prototypes that can represent the "spatial" distributions in the dataset by identifying the most centralised objects - that can attract a large number of objects - at a given input parameter value δ_o. A hyperspherical region of radius δ_o is defined as the neighbourhood of object O_i, NO_i, and the total number of neighbouring objects within this region, $W(O_i)$, is considered as a weight for this object.

Prototypes generation can be summarised in the following steps:

1. Choose the object with maximum weight, O_M, and all objects $O_{M1}, \ldots, O_{Mj}$ within its neighbourhood NO_M and find their corresponding neighbourhoods $NO_{M1}, \ldots, NO_{Mj}$.

2. Find the intersection between neighbourhood of O_M and neighbourhoods of its closest objects $O_{M1}, \ldots, O_{Mj}$ as:

$$N_{INT} = NO_M \cap NO_{M1} \cap NO_{M2} \ldots \cap NO_{Mj}$$

3. Define "prototype" B_k as the object (or mean of objects) that results from the intersection operation.

4. Clear all weights for O_M and $O_{M1}, \ldots, O_{Mj}$ to avoid possibility of generating more than one prototype within δ_o. This allows the possibility of generating prototypes in the sparsely populated areas, where the objects will have lower weights.

5. The process of generating prototypes is continued until no more weighted object is found.

After generating proper prototypes (K prototypes), the clustering problem is reduced to assigning the n objects to the nearest of K prototypes to create K clusters. The prototypes (cluster centres) are subsequently updated to the mean of their assigned objects. This process is repeated until no more changes occur in the prototypes. In order to achieve more compact clusters and yet with wider separations between clusters, i.e., better clustering quality, *RACAL* is enhanced with a reliable validity index to evaluate the clustering result at each update of the prototypes. The best clustering result with a given δ_o is the one that

achieves a maximum value of the validity index. Therefore, the enhancement of the clustering function with a reliable validation index can be used to produce the best results for a given δ_o value. The cluster validity index "I" [59] has been selected from a number of available validity indices (see Section 3.3.1). The objective is to maximise the "I" index for achieving proper clustering with a given δ_o value. The proposed clustering algorithm (*RACAL*) is detailed in Algorithm 2.

4.3 Datasets

Five different types of real-world data are used to investigate whether the proposed algorithm scales well with the size and dimension of the dataset or not.

4.3.1 Communication data

Two communications datasets with different noise conditions have been used to demonstrate the clustering behaviour of *RACAL*, namely, QAM-4 and QAM-16 at SNR = 15 and SNR = 10 dB (see Section 3.2.1).

4.3.2 Breast cancer data

Two Wisconsin breast cancer datasets [76] have been used in this study (see Section 3.2.2).

4.3.3 Leukaemia data

This dataset [82] consists of 72 bone marrow samples over 7,129 probes (which are treated as features) from 6,817 human genes, and contains two classes: class 1 (25 AML "Acute Myeloid Leukaemia" samples) and class 2 (47 ALL "Acute Lymphoblastic Leukaemia" samples).

4.3.4 Iris data

This dataset is one of the best known dataset found in pattern recognition literature [76]. The dataset consists of three classes of fifty examples and each with four features, where each class refers to a type of iris plant namely Iris Setosa, Iris Versicolor, Iris Verginica.

Algorithm 2 The proposed clustering algorithm (*RACAL*)

1. Read the input data $O = \{O_1, \ldots, O_n\}$ and the radius value δ_o.
2. Calculate the distance matrix D and its corresponding relational matrix "normalised distance matrix" $R = [r_{ij}]$, and $r_{ij} \in [0, 1]$.
3. For each object O_i. Identify the neighbouring objects NO_i according to the value of δ_o, and its corresponding weight $W(O_i)$.

$NO_i = (r_{ij} \leq \delta_o)$; # (bi-valued vector)
$W(O_i) = sum(NO_i == 1)$;

let $K = 1$.

4. **Do**

Find the max weighted object & clear the weights of its neighbouring objects

$W_{max} = \max_{i=1,\ldots,n} W(O_i)$;
$W(NO_M == 1)) = 0$;

#Find the intersection between neighbourhood of O_M and neighbourhoods of its neighbouring objects $O_{M1}, \ldots, O_{Mj}$

$N_{INT} = NO_M \cap NO_{M1} \cap NO_{M2} \ldots \cap N_{Mj}$ # (bi-valued vector)

Generate a prototype

$B_k(K) = mean(O(N_{INT} == 1, :))$;
$K = K + 1$;

Until no weights.

let $V_{prev} = 0$.

5. For each object O_i

For each prototype B_k
IF ($d(O_i, B_k) \leq \delta_o$) break;
Else Continue;
EndFor
IF no prototype found
Generate this object as a new prototype
Else
Continue;
EndFor

6. Let each object select the nearest prototype.
7. Calculate the validation value $V_i = I(K)$ of the clustered data.

IF ($V_i > V_{prev}$)
CR = the clustering result of V_i. # (partition vector)
$V_{prev} = V_i$;
EndIF

8. Repeat step (5) until no change.
9. Save the clustering result (CR) of the best validation value.

The first class is linearly separable from others while the other two classes are not linearly separable. The measurement consists of the sepal and petal lengths and widths in cms.

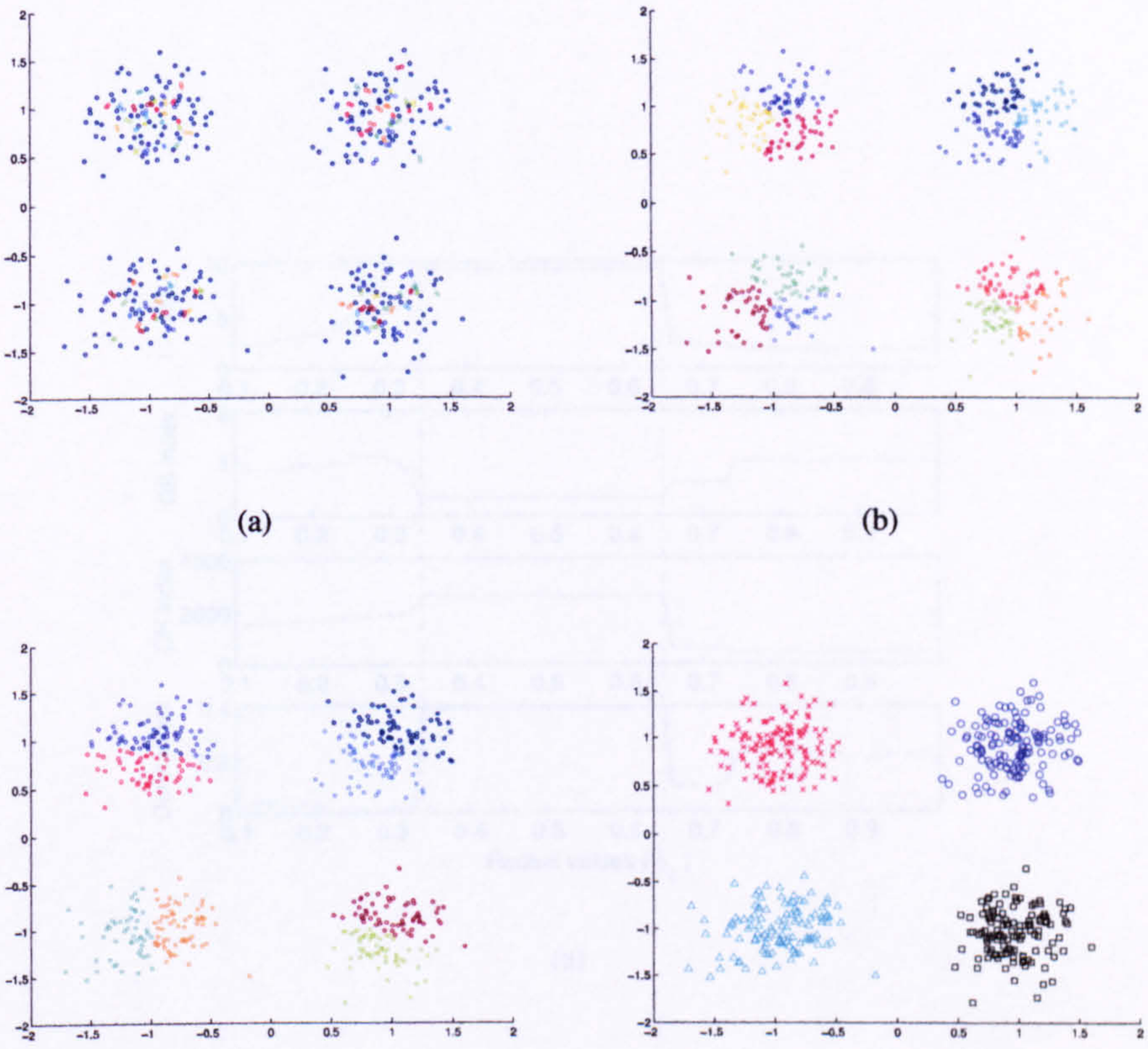
4.3.5 STARE data

STARE dataset is a publicly available dataset [83]. The dataset consists of 20 images which are digitised slides captured by a TopCon TRV-50 fundus camera at 35° FOV. Each slide was digitised to produce a 605×700 pixels image, standard RGB, 8 bits per colour channel. Every image has been manually segmented by two observers to produce ground truth vessels segmentation. In this application, each object "pixel" is characterised by three features [84] and the dataset corresponds to the set of all 423,500 (605 × 700) pixels, each with its three feature values.

4.4 Clustering Results

4.4.1 Communication datasets

A series of experiments at $\delta_o = 0.01, ..., 0.99$ were carried out to investigate the clustering behaviour of the proposed algorithm. Figure 4.1 shows the clustering performance on a QAM-4 signal at SNR = 15 dB for four different values of δ_o. Each colour on the plot shows different clusters. At $\delta_o = 0.01$, the number of clustered objects are 264 out of 512 objects with 117 clusters (the blue colour in Fig. 4.1(a) represents the unclustered objects). The performance is very poor, because the radius value is insufficient to capture the whole dataset. As the value of δ_o is increased, the spread around each cluster increases, increasing the membership of each cluster, and encompassing more of the data. At $\delta_o = 0.2$, 0.28, and 0.4, all the data objects are being clustered to 12, 8, and 4 clusters respectively. Finally, the validation indices, *I* [59], *DB* [56], *Dunn* [57], and *CH* [58], are used to evaluate these different results. Figure 4.2 contains two plots: part (a) shows the validation indices values versus the radius (δ_o), and part (b) shows the number of clusters K corresponding to the same δ_o values. As shown in Fig. 4.2, the corresponding number of clusters K to the δ_o = [0.34 : 0.64] for which *I*, *CH*, *Dunn* validation indices are maximised and *DB* validation

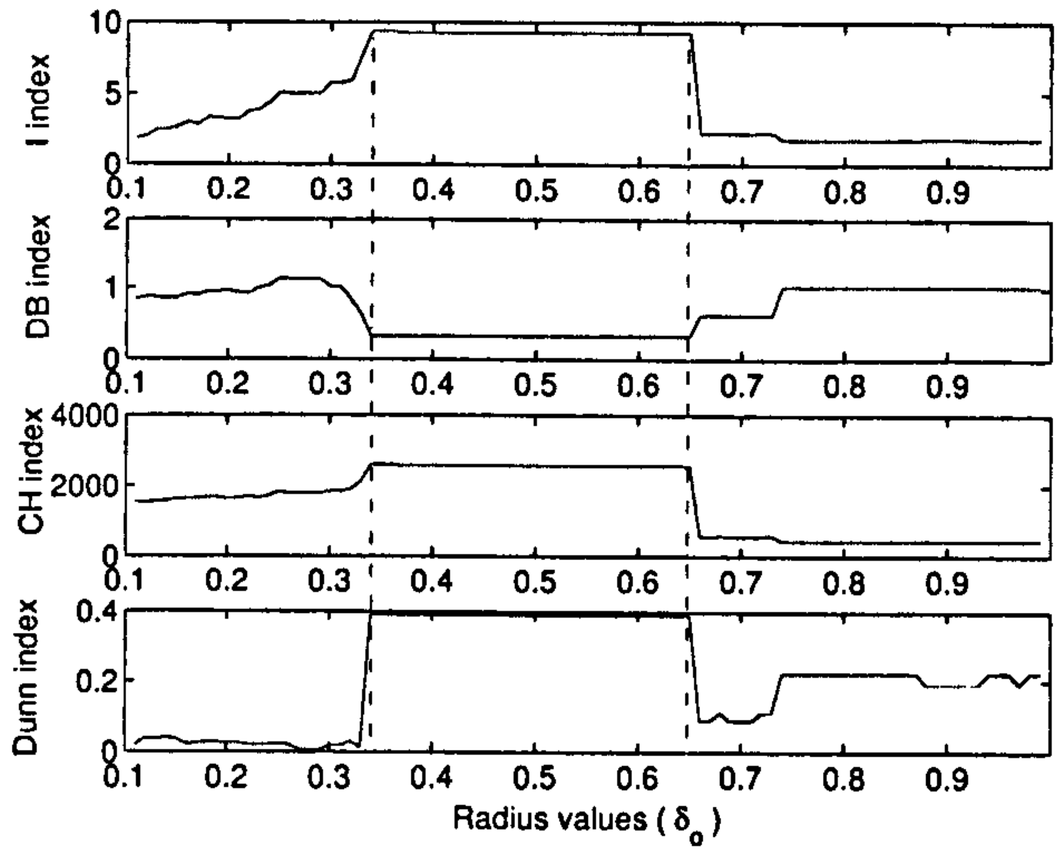


(a) (b) (c) (d)

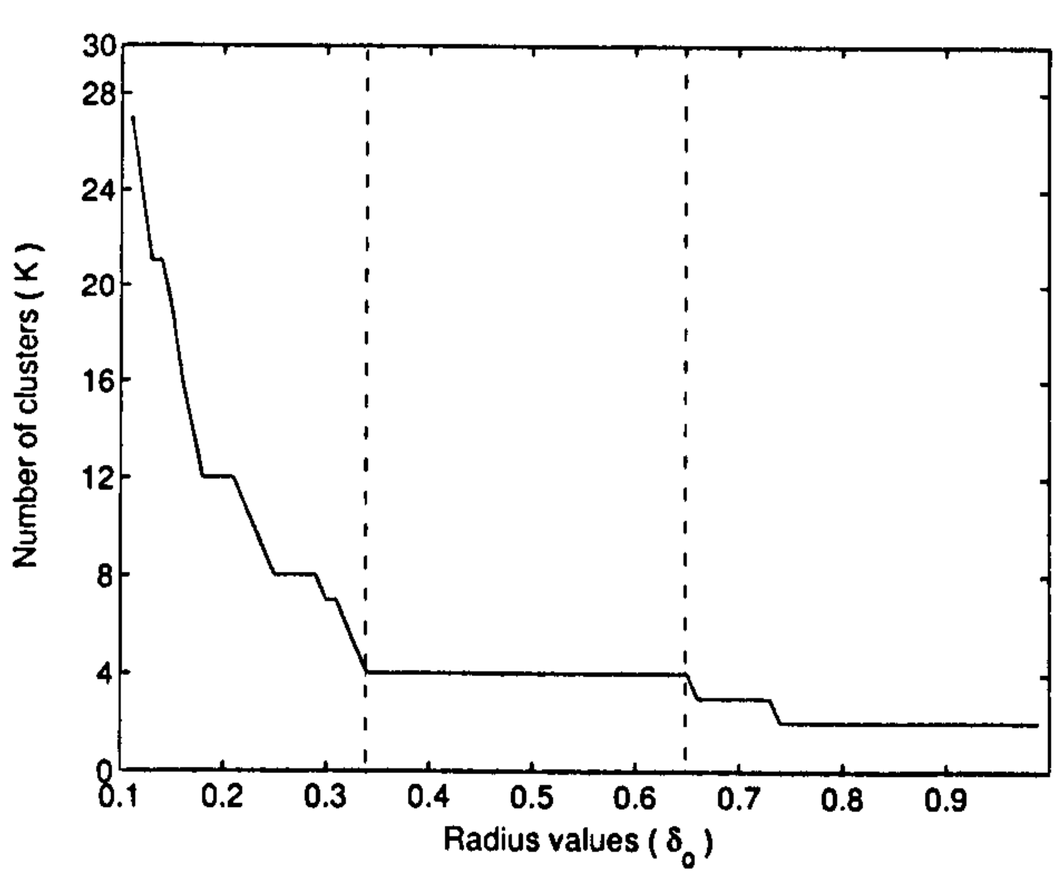
Figure 4.1: Clustering performance for QAM-4 at SNR = 15 dB, for (a) $\delta_o = 0.01$ "the blue colour represents the unclustered objects", (b) $\delta_o = 0.2$, (c) $\delta_o = 0.28$, and (d) $\delta_o = 0.4$.

index is minimised, is $K = 4$. The value of $K = 4$ represents the best fit number of clusters which actually coincides with the true number of clusters.

Communications data with SNR = 15 dB or higher is relatively easy to cluster; however, data with SNR = 10 dB is much harder, as the definition between clusters is much less clear than the higher SNR values. Figure 4.3 shows the clustering performance on QAM-4 at SNR = 10 dB for four different values of δ_o. This shows that as the value of δ_o is increased, the clustering becomes consistent and relatively well distributed through the data space. As δ_o is increased, all clusters would gradually merge to form one large cluster. As shown in Fig. 4.4, all the selected validation indices indicate $K = 4$ as the best fit number of clusters which actually coincides with the true number of clusters. The flat region in the plots of Fig. 4.4 within the range $\delta_o \in [0.48 : 0.62]$ is narrower than the flat region of SNR = 15

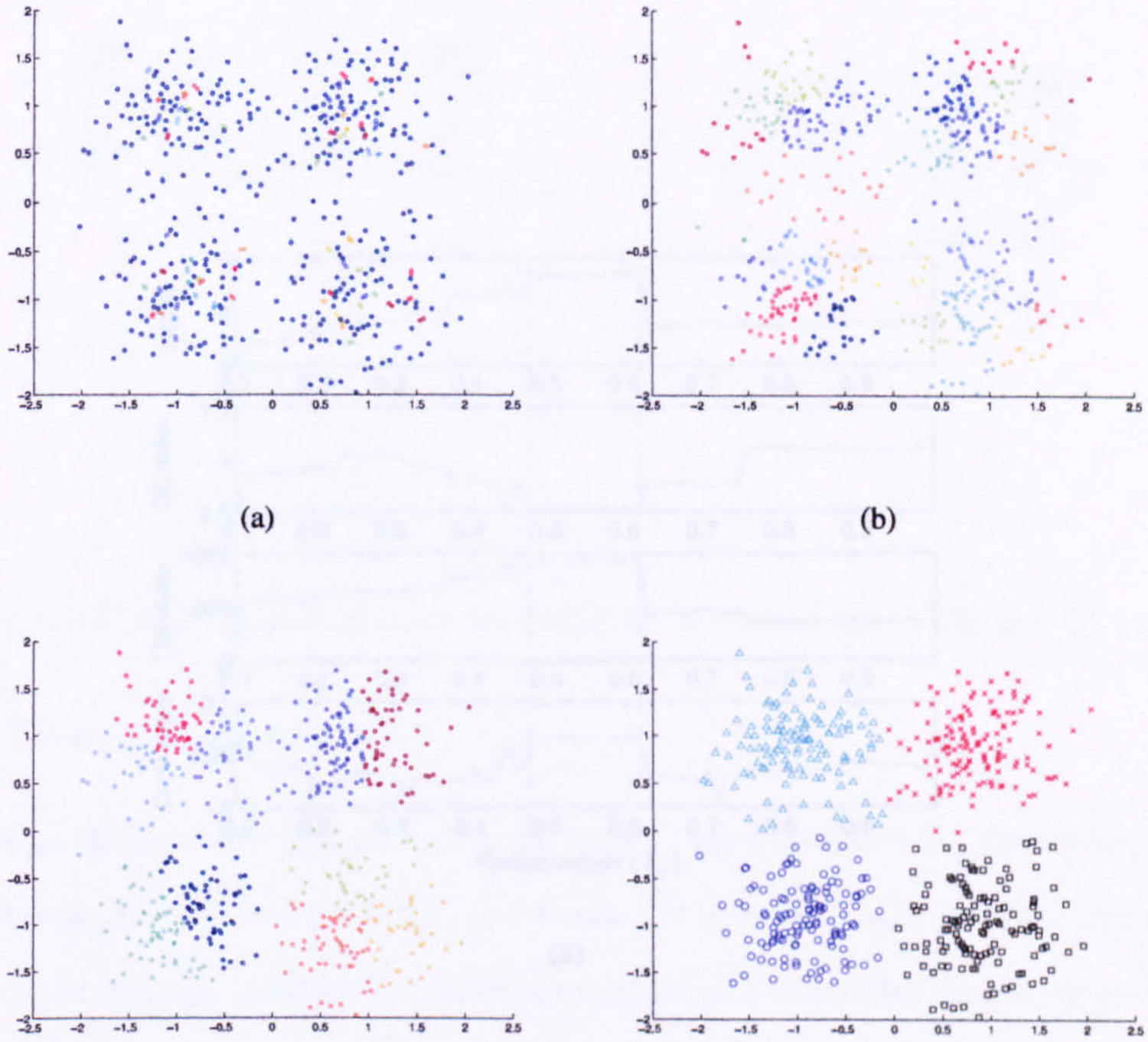


(a)



(b)

Figure 4.2: (a) Evaluations for different δ_o values on QAM-4 at SNR = 15 dB, and (b) the number of clusters versus the radius δ_o.



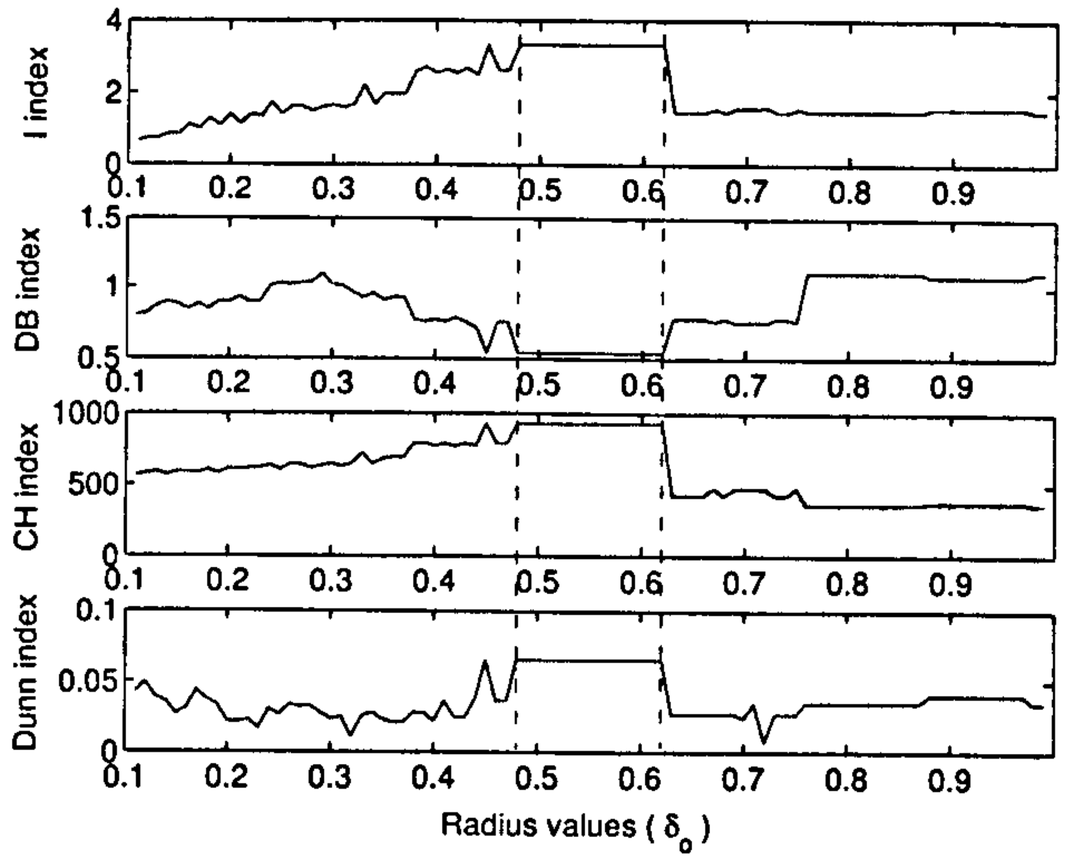
(a) (b)

(c) (d)

Figure 4.3: Clustering performance for QAM-4 at SNR = 10 dB, for (a) $\delta_o = 0.01$ "the blue colour represents the unclustered objects", (b) $\delta_o = 0.12$, (c) $\delta_o = 0.3$, and (d) $\delta_o = 0.48$.

dB within the range $\delta_o \in [0.34 : 0.64]$, as shown in Fig. 4.2. This is because of the low separation between clusters.

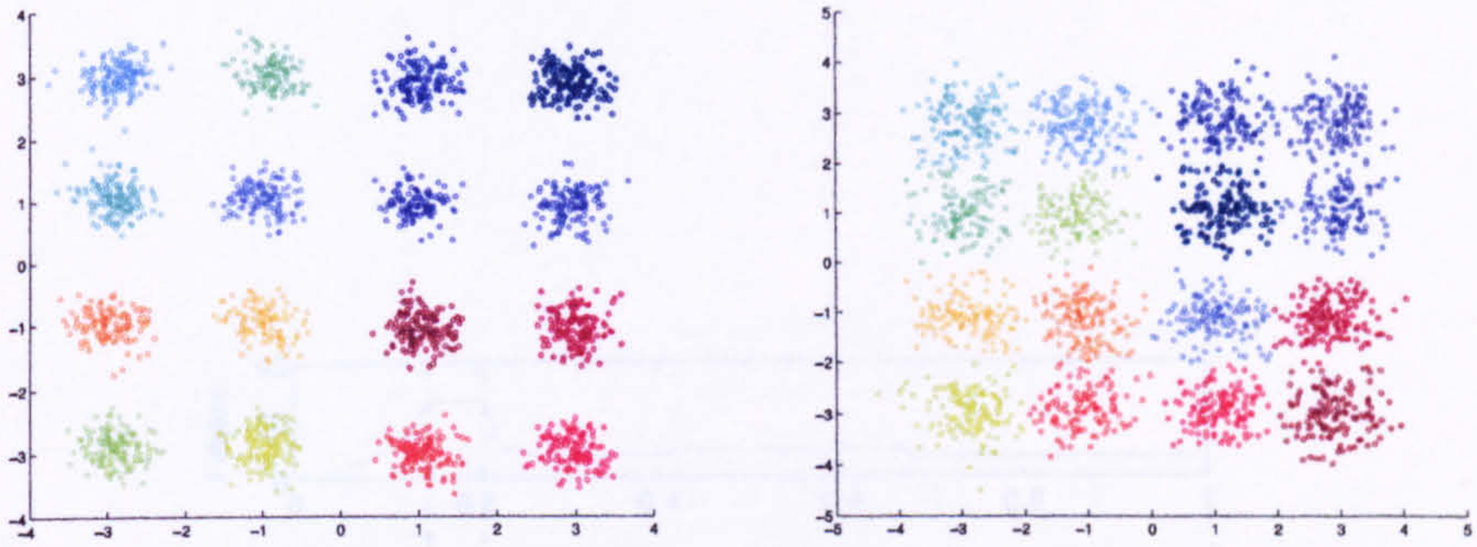
Similar experiments were carried out on QAM-16 at SNR = 15 dB and SNR = 10 dB. Figure 4.5 shows the clustering performance on a QAM-16 at SNR = 15 dB, $\delta_o = 0.18$ for producing 16 clusters and the clustering performance at 10 dB SNR, $\delta_o = 0.20$ for producing 16 clusters. Figure 4.6 shows the validation indices values against the input parameter δ_o for QAM-16 at SNR = 15 dB. As shown, the number of clusters K for which *I*, *CH*, *Dunn* validation indices are maximised, and *DB* validation index is minimised is now $K = 16$, which actually coincides with the true number of clusters.



(a)

(b)

Figure 4.4: (a) Evaluations for different δ_o values on QAM-4 at SNR = 10 dB, and (b) the number of clusters versus the radius δ_o.



(a) (b)

Figure 4.5: Clustering performance for QAM-16 at SNR of (a) 15 dB, and (b) 10 dB.

4.4.2 Breast cancer datasets

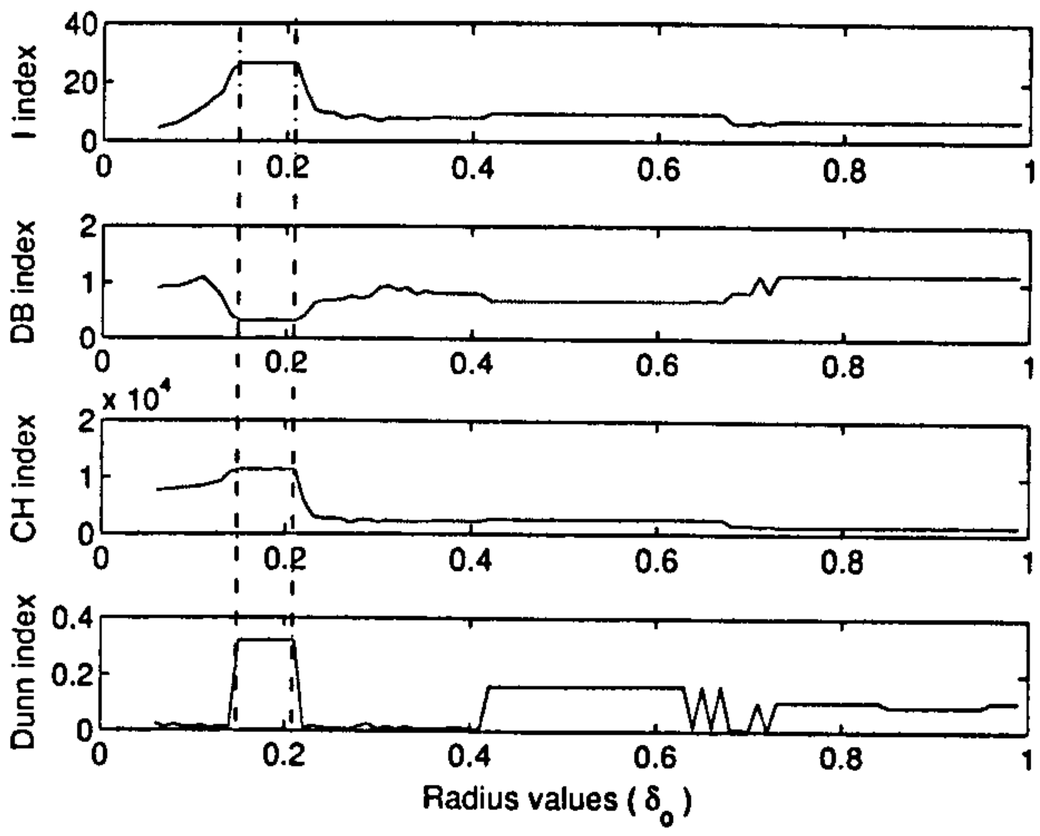
Similar experiments were carried out on the breast cancer datasets. As shown in Fig. 4.7, within the range $\delta_o \in [0.69 : 0.99]$, *I*, *DB* and *CH* indices clearly indicate two clusters as the best fit clusters, however the maximum values of *Dunn* index do not agree with the above conclusion, because of its sensitivity to the presence of noise in the datasets [7, 55]. As shown in Table 4.1, the clustering performance at $K = 2$ (true number of clusters) for dataset 1 and 2 is 96.4 % and 92.7 % respectively.

4.4.3 Leukaemia dataset

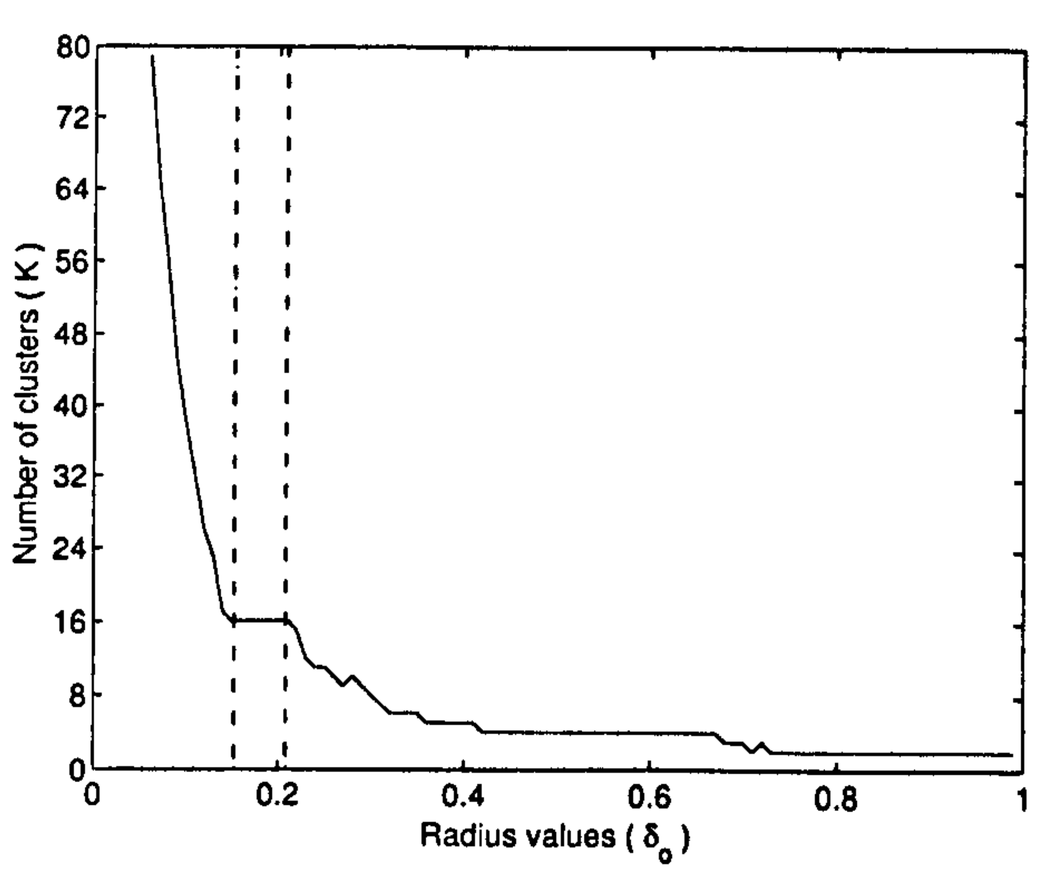
Similarly, when *RACAL* is applied to cluster leukaemia dataset, two clusters are chosen by *I*, *DB* and *CH* indices as the best fit clusters which coincides with the true number of clusters underlying the dataset. Figure 4.8 shows the clustering performance of dataset at different δ_o values.

4.4.4 Iris dataset

Similarly, Fig. 4.9 shows the clustering performance of *RACAL* when applied to cluster Iris data at different δ_o values. As shown, three clusters are chosen by *I*, *CH* and *Dunn* indices as the best fit clusters which coincides with the true number of clusters underlying the dataset; however, *DB* index chooses two clusters as the best fit clusters, where its between-

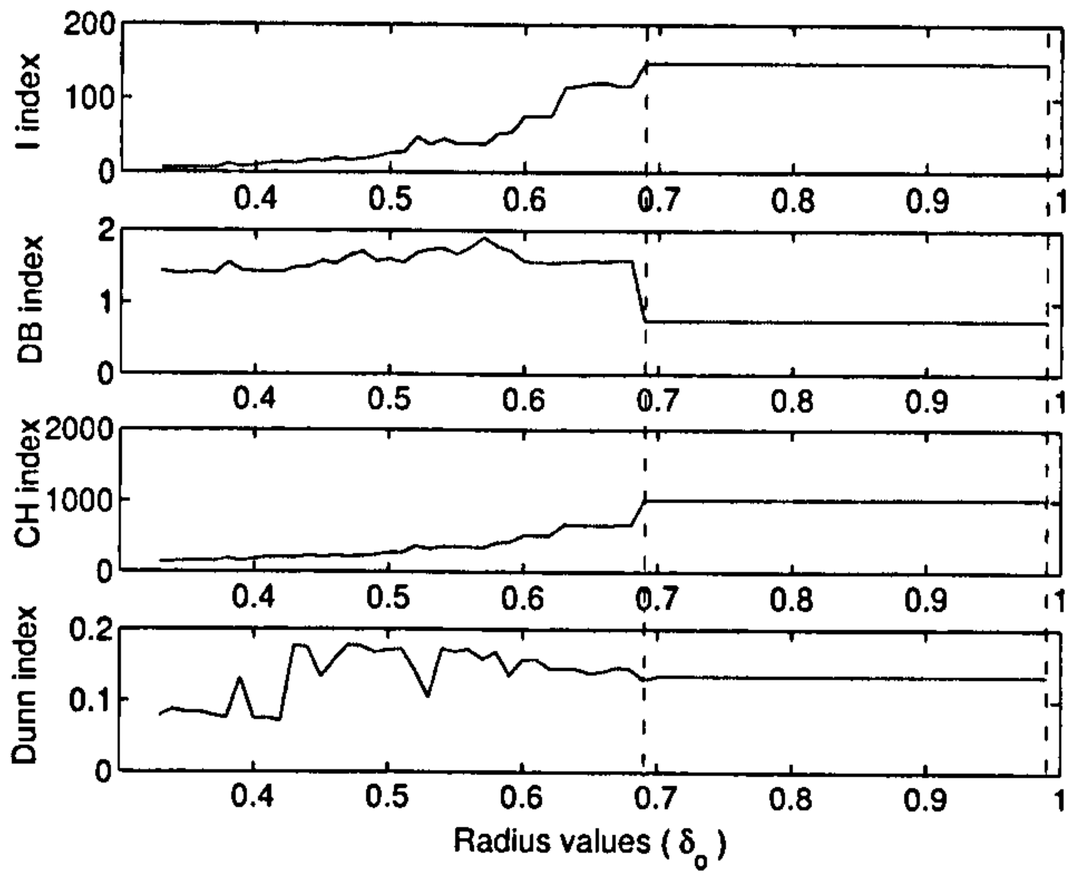


(a)

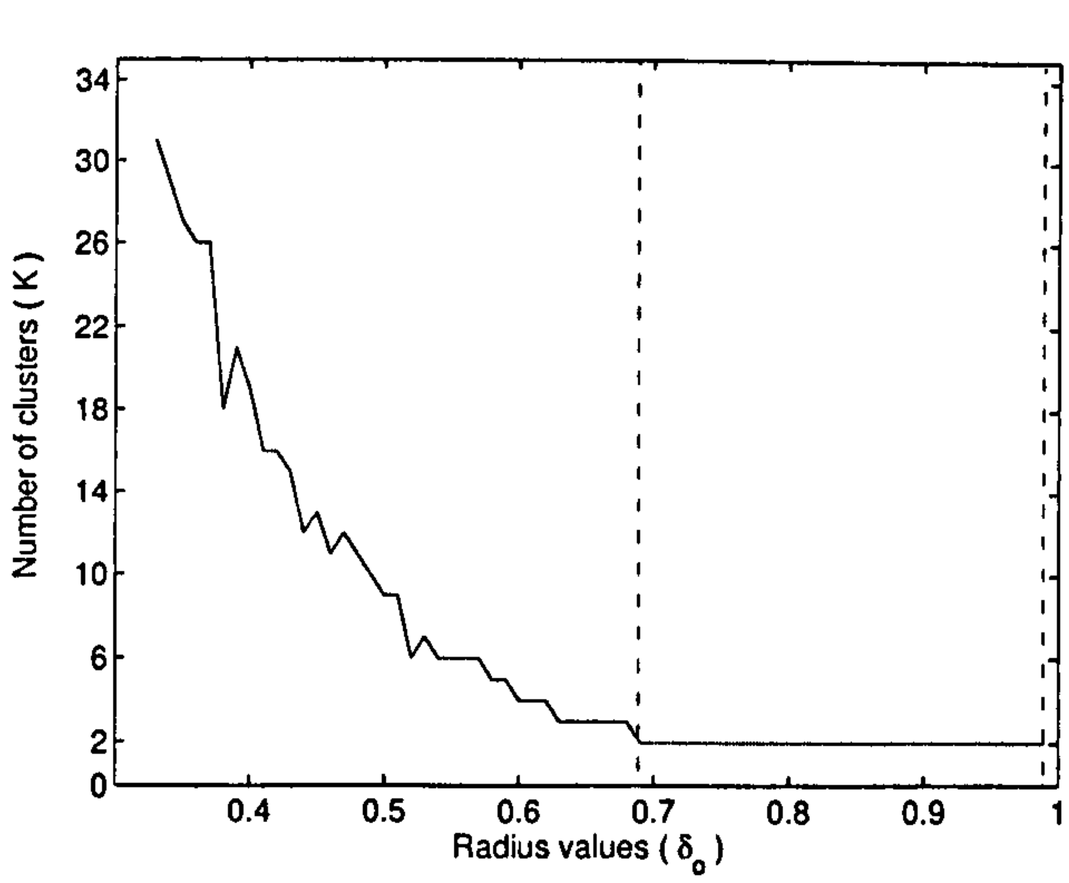


(b)

Figure 4.6: (a) Evaluations for different δ_o values on QAM-16 at SNR = 15 dB, and (b) the number of clusters versus the radius δ_o.

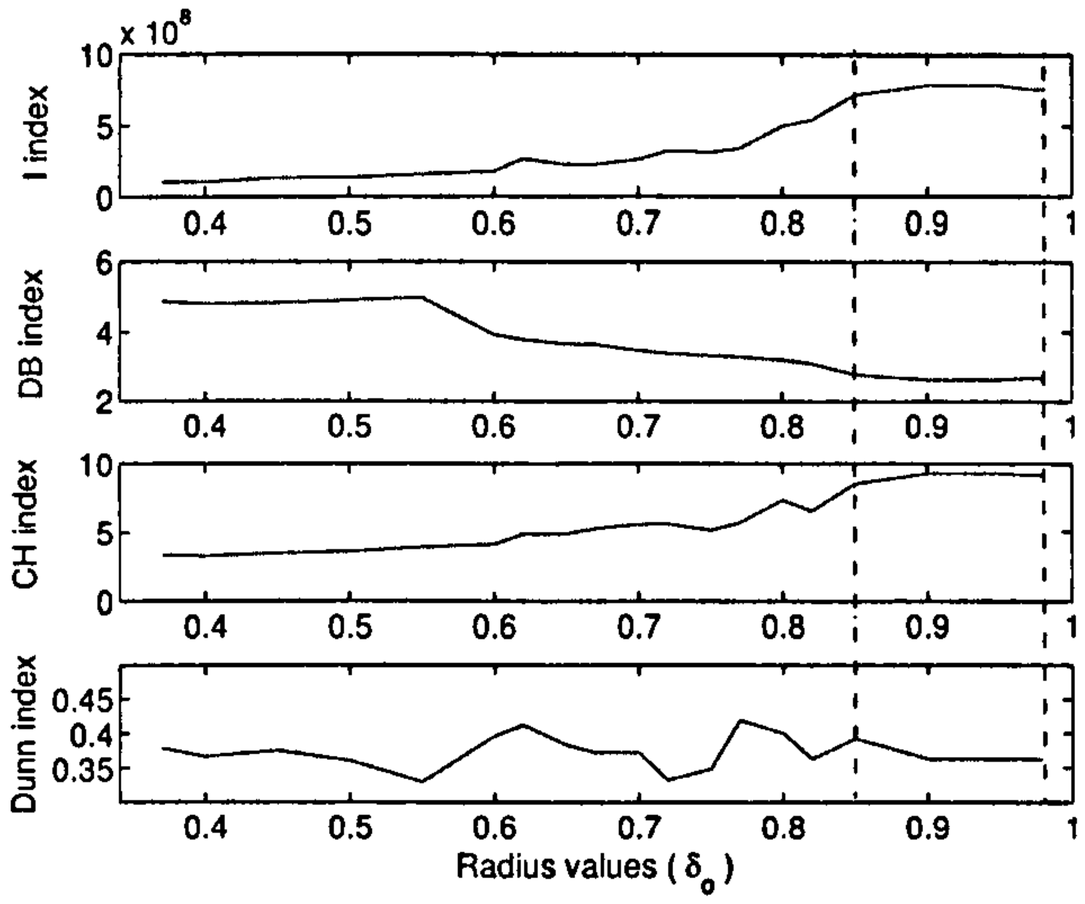


(a)

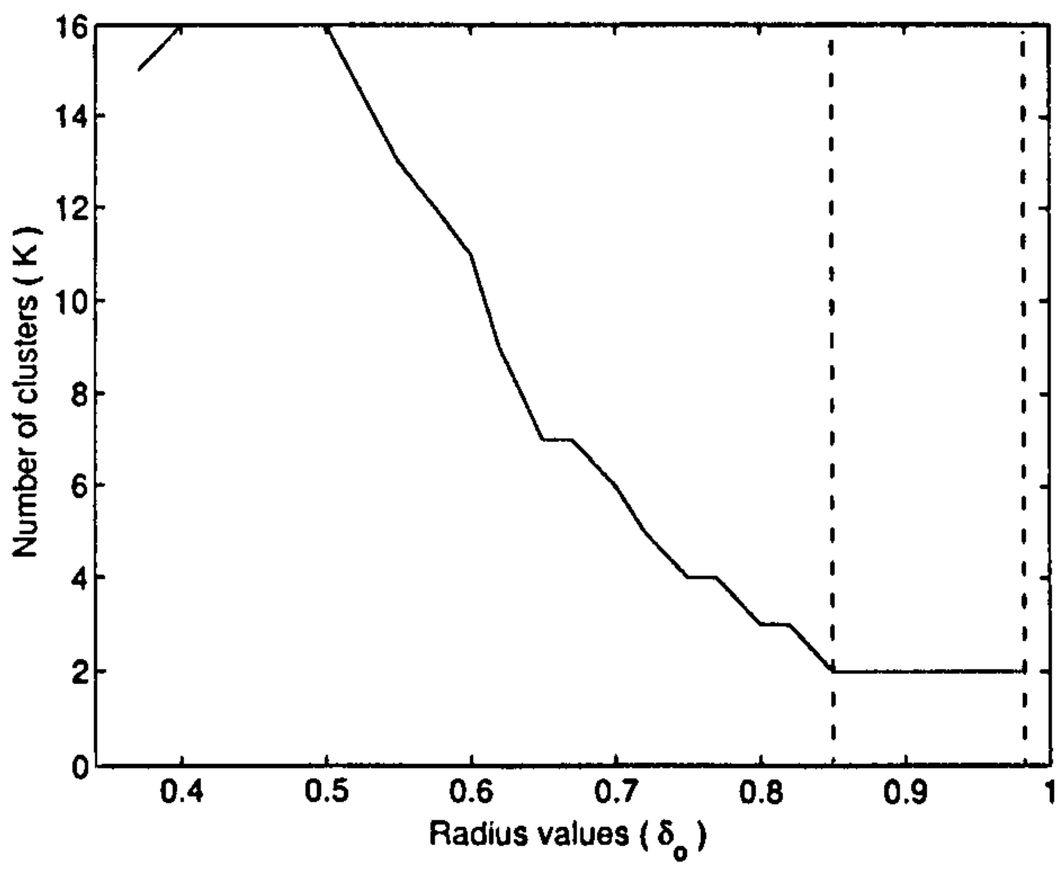


(b)

Figure 4.7: (a) Evaluations for different δ_o values on breast cancer dataset 1, and (b) the number of clusters versus the radius δ_o.

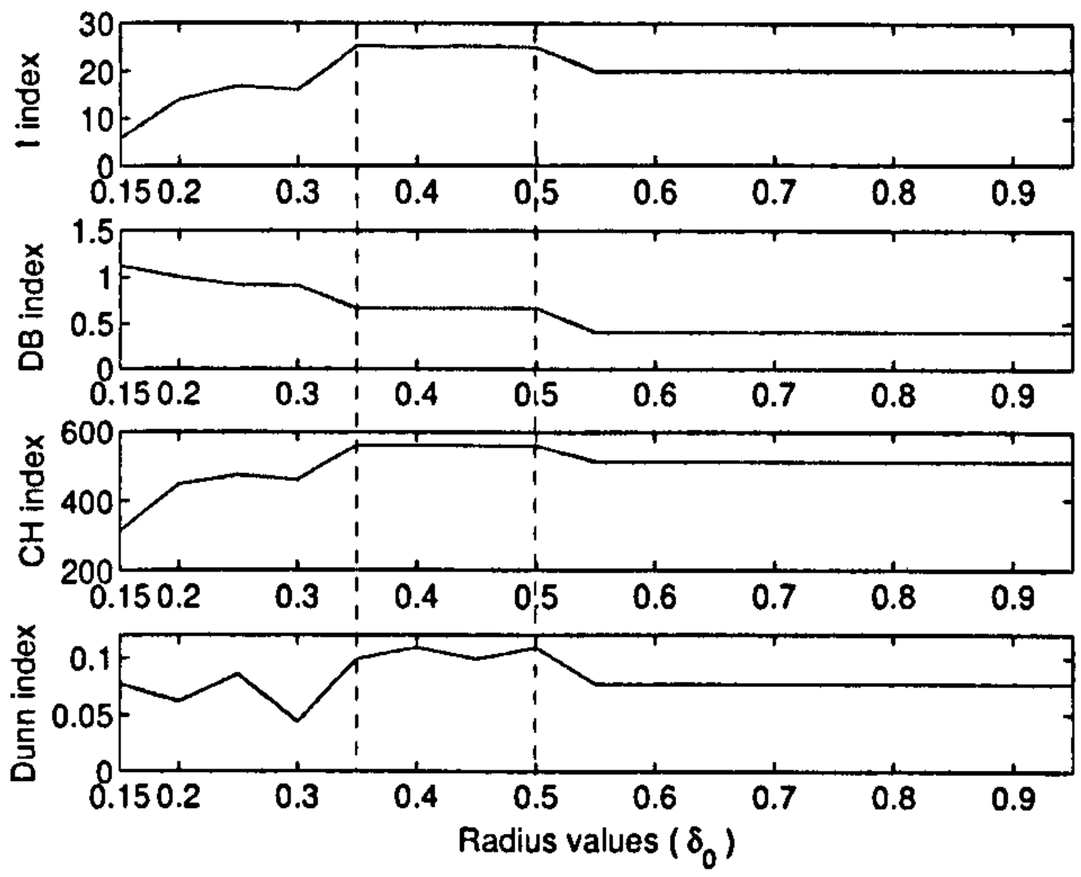


(a)

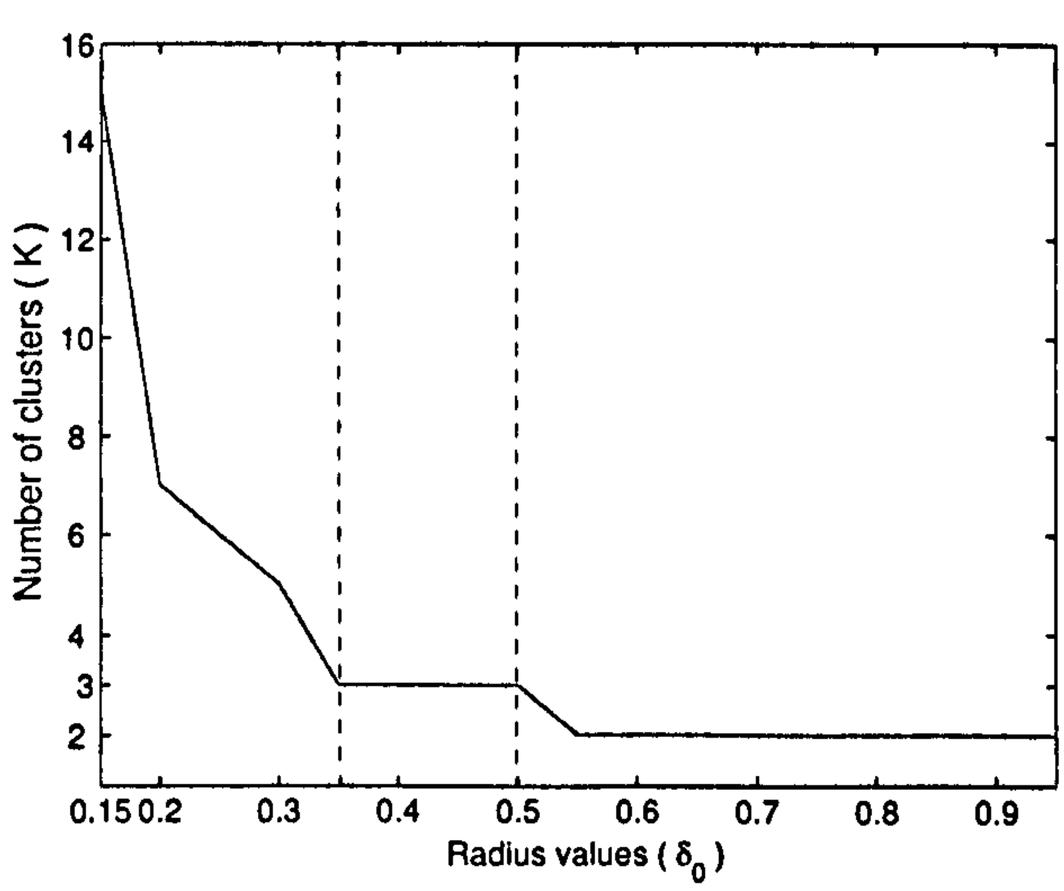


(b)

Figure 4.8: (a) Evaluations for different δ_o values on leukaemia dataset, and (b) the number of clusters versus the radius δ_o.



(a)



(b)

Figure 4.9: (a) Evaluations for different δ_o values on iris dataset, and (b) the number of clusters versus the radius δ_o.

Table 4.1: Clustering accuracies comparison, based on 100 experiments.

	d	K-means	SOM	K-medoids	CLARA	RACAL
QAM-4 (15 dB)	2	**100 %**	99.8 %	**100 %**	**100 %**	**100 %**
QAM-4 (10 dB)	2	91.3 %	96.1 %	**100 %**	98.33 %	**100 %**
QAM-16 (15 dB)	2	90.26 %	91.25 %	**100 %**	92.76 %	**100 %**
QAM-16 (10 dB)	2	88.21 %	82.26 %	99.85 %	92.48 %	**100 %**
Breast cancer 1	9	96.2 %	**97.2 %**	95.9 %	95.34 %	96.4 %
Breast cancer 2	30	91.1 %	91.6 %	89.1 %	90.61 %	**92.7 %**
Leukaemia	7,129	63.3 %	70.5 %	60.3 %	60.7 %	**72.2 %**
Iris	4	83.2 %	86.4 %	90.1 %	89.8 %	**92.6 %**

where **d** is the number of dimensions

cluster term outpaces the decreasing rate of the within-cluster term [85]. This indicates the importance of using different validation methods for estimating the number of clusters.

4.5 Comparison with other Clustering Algorithms

For purposes of comparison, the performance of the proposed *RACAL* algorithm is compared with other clustering algorithms: K-means [1, 7], SOM [29], K-medoids [6], and CLARA [6, 18] algorithms. As shown in Table 4.1, the performance of the proposed *RACAL* clustering algorithm is more efficient and able to find clustering results of better accuracies than those found by other algorithms mentioned above. In the following, extensive comparisons based on the clustering performance measures (*CPMs*) are carried out (see Chapter 3), where the performance of the proposed clustering algorithm, *RACAL*, is evaluated at different number of clusters which are produced by varying the radius parameter δ_o ($\delta_o = [0 : 1]$). This process is repeated one hundred times using sub-sampling approach. The number of clusters K that produced by the radius parameter δ_o using the proposed *RACAL* clustering algorithm is used to partition the dataset by the other partitional clustering algorithms that are considered in this study. Therefore, one can get a fair comparison. The four Figures 4.10, 4.11, 4.12, and 4.13 contain two plots each. The subplot

on top shows the mean validation index (of 100 runs) of each clustering algorithm versus the number of clusters (K), where higher validation values give better robustness. The lower subplot shows the repeatability of clustering algorithms over the same K values. The number of clusters K corresponding to the higher validation and repeatability values are expected to be the true number of clusters that underlying the dataset. As shown in Tables 4.2, 4.3, 4.4 and 4.5, *RACAL* possesses higher robustness as well as offers better repeatability compared with other clustering algorithms. Also, the performance of K-medoids algorithm is better than the performance obtained by K-means, SOM and CLARA algorithms. However, K-medoids algorithm is not as efficient computationally as others, as its computational time is higher than other algorithms.

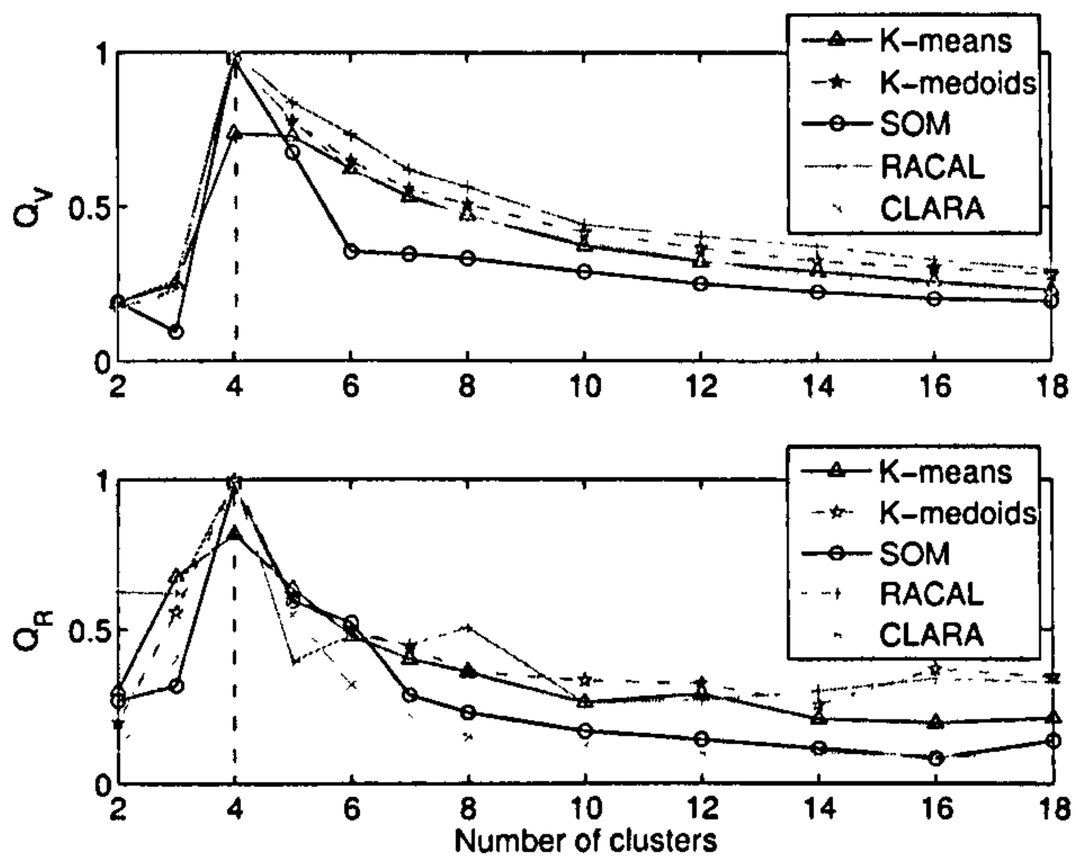


Figure 4.10: Testing rule procedure on QAM-4 at SNR = 15 dB.

Table 4.2: Performance evaluation results on QAM-4 at SNR = 15 dB.

	SOM	K-means	K-medoids	CLARA	RACAL
$< Repeatability >$	0.32	0.40	0.44	0.27	0.46
$< Validation >$ ("I" index)	0.35	0.42	0.46	0.44	0.50
CPM_1	6.4	6.6	6.9	6.5	7.1
CPM_2	0.55	0.65	0.70	0.59	0.74

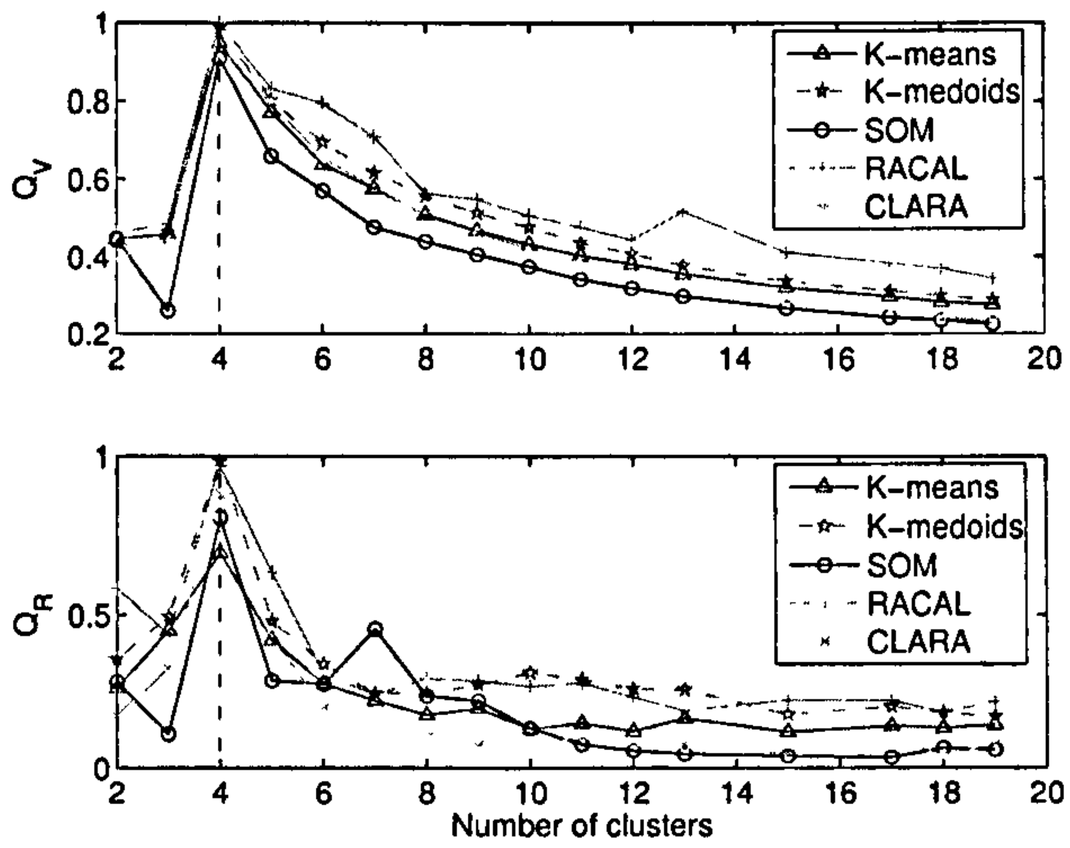


Figure 4.11: Testing rule procedure on QAM-4 at SNR = 10 dB.

Table 4.3: Performance evaluation results on QAM-4 at SNR = 10 dB.

	SOM	K-means	K-medoids	CLARA	RACAL
$< Repeatability >$	0.20	0.24	0.33	0.19	0.35
$< Validation >$ ("I" index)	0.40	0.47	0.50	0.46	0.55
CPM_1	8.18	8.41	8.94	8.31	9.16
CPM_2	0.52	0.59	0.66	0.56	0.71

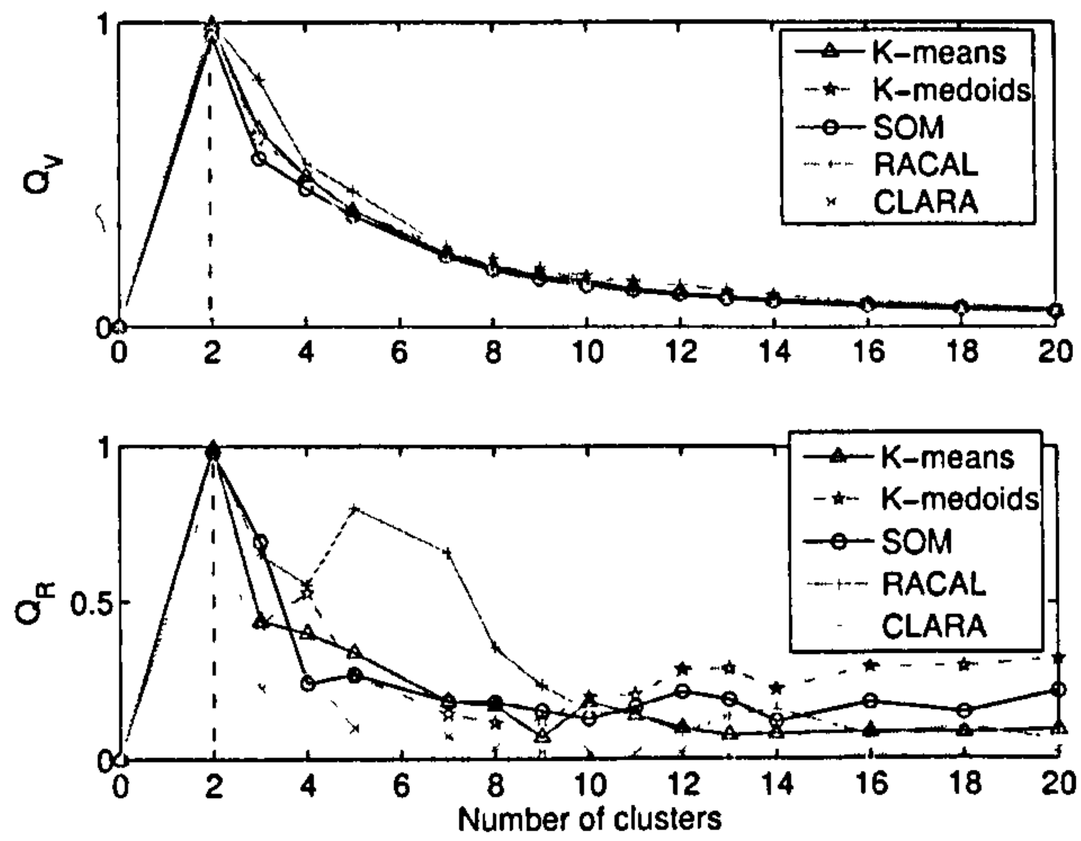


Figure 4.12: Testing rule procedure on breast cancer dataset.

Table 4.4: Performance evaluation results on breast cancer dataset.

	SOM	K-means	K-medoids	CLARA	RACAL
$< Repeatability >$	0.27	0.23	0.31	0.12	0.34
$< Validation >$ ("I" index)	0.24	0.26	0.27	0.25	0.29
CPM_1	8.02	7.95	8.19	7.61	8.41
CPM_2	0.45	0.43	0.49	0.34	0.53

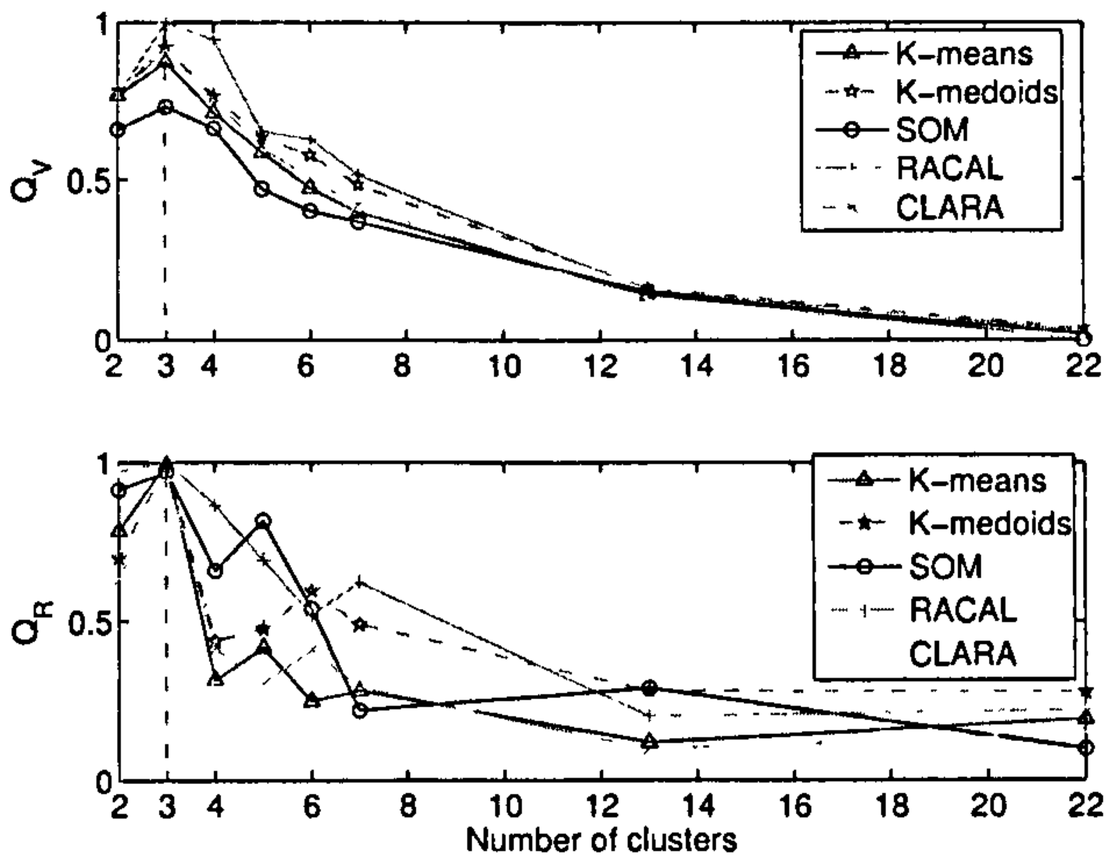


Figure 4.13: Testing rule procedure on Iris dataset.

Table 4.5: Performance evaluation results on Iris dataset.

	SOM	K-means	K-medoids	CLARA	RACAL
$< Repeatability >$	0.56	0.42	0.53	0.41	0.55
$< Validation >$ ("I" index)	0.43	0.49	0.54	0.51	0.58
CPM_1	4.81	4.73	5.02	4.74	5.31
CPM_2	0.75	0.70	0.78	0.71	0.81

4.6 RACAL with Partial Supervision Strategy (RACAL-PS)

In this section, adaptation of *RACAL* algorithm with some labelled objects to guide the clustering process of the unlabelled objects, i.e. *RACAL* with partial supervision (*RACAL-PS*), is proposed. The proposed method is divided into two stages. First stage is to cluster n objects (the entire objects) into K clusters according to the input parameter δ_o, as described in Section 4.2. Therefore, K clusters can be obtained according to the input parameter δ_o. Second stage is to randomly select N_P objects from the dataset to be labelled data objects, then each cluster is classified according to the class label of the majority of its objects (the choice of N_P and δ_o ensure that all clusters can be classified). For each labelled data object, if its cluster is classified to different class (label), then this data object will be assigned to the nearest cluster that has the same label as it. Once the clusters' memberships are updated, the cluster centres can be updated, and the process continues checking the cluster memberships of the labelled data objects until no changes. Then, all the data objects that belong to different clusters with the same class labels can be assigned to that label. This proposed method will bias clustering towards a better search space. The proposed supervised method is detailed in Algorithm 3.

A soft classification, where all objects are allowed in principle to belong to all classes with different degrees of membership, is achieved by adding the fuzzy memberships for each object with the clusters that belong to the same class label. Eqs. 4.1 and 4.2 show the fuzzy membership (m_{ik}) of object k to cluster i, and the soft membership ($M_{C_i k}$) of object k to class C_i respectively.

$$m_{ik} = \frac{1}{\sum_{j=1}^{K} \left(\frac{d_{ik}}{d_{jk}}\right)^{2/(q-1)}} \tag{4.1}$$

$$M_{C_i k} = \sum_{j}^{K} m_{jk} \quad if\ cluster\ j \in class\ C_i \tag{4.2}$$

where q is the fuzziness exponent, d_{ik} is the distance from object k to the current cluster

Algorithm 3 *RACAL* with partial supervision strategy (*RACAL-PS*)

- *Step 1: Cluster the dataset using RACAL algorithm to find the proper prototypes.*
- *Step 2: Apply the supervision strategy as follow:*

  1. Randomly select N_P objects from the dataset to be labelled data objects.
  2. Classify the clusters obtained by *RACAL* algorithm to the class of its most labelled data objects.
  3. For each labelled data objects, if its cluster is classified to different class (label), then this data object will be assigned to the nearest cluster that has the same label as it. Otherwise, assign it to the nearest unclassified (unlabelled) cluster.
  4. Update the clusters' memberships and the cluster centres.
  5. goto (2) until no change in the clusters' memberships.

---

centre i, and d_{jk} is the distance from object k and any of cluster centre j ($1 \leq j \leq K$).

4.6.1 Classification results

A series of experiments were carried out to examine the performance of *RACAL* when applying the proposed supervision strategy, where there are some labelled objects. In these experiments, effects of the number of labelled objects on the classification accuracy are investigated. Fractions of objects from the entire dataset are randomly selected to be used as labelled objects. For each fraction, this process is repeated one hundred times without replacement. The best, average, and standard deviation of classification accuracy are obtained over one hundred runs for each fraction of labelled objects. For breast cancer dataset 1 as demonstrated in Table 4.6, the best classification accuracy increases initially with the increasing fraction of the labelled objects and then settled down. By examining the average and standard deviation of the classification performance, when 5% of the entire dataset are labelled, the average performance is being the lowest and it has the highest standard deviation compared with the other fractions of labelled objects. For more than 5% labelled objects, the average classification accuracy is higher than 97% with comparable best accuracies and small magnitude of standard deviation as demonstrated in Table 4.6. For breast cancer dataset 2 as demonstrated in Table 4.7, standard deviations are higher than standard deviations of breast cancer dataset 1. This is because of the lower compactness of clus-

ters on breast cancer dataset 2 compared with breast cancer dataset 1, as indicated in [86]. As demonstrated in Table 4.7, the best and average classification accuracy of 98.3% and 95.2% respectively were achieved at 30% labelled objects, with the lowest being 95.8% and 90.6% for best and average accuracies respectively at 5% labelled objects. Similarly, for leukaemia dataset as demonstrated in Table 4.8, it can be observed that using 20% labelled objects achieves average performance of 89% which is 9% higher than the average performance when using 10% labelled objects, while the standard deviation of classification accuracy when using 20% labelled objects is 6% less than the standard deviation of classification when using 10% labelled objects. In case of Iris dataset, as demonstrated in Table 4.9, the best and average classification accuracy of 99.3% and 97.1% respectively were achieved at 30% labelled objects, with the lowest being 97.1% and 95.8% for best and average accuracies respectively at 10% labelled objects. With the increase of labelled objects percentage, the overall performance increases, i.e. higher average and best classification accuracy with smaller magnitude of standard deviation.

Table 4.6: Classification accuracy (%) for breast cancer dataset 1 using *RACAL-PS*, based on 100 experiments.

labelled objects %	Best (%)	Average (%)	Std (%)
5	97.5	96.6	1.4
10	97.9	97.2	0.3
15	98.2	97.4	0.3
20	98.6	97.6	0.3
25	98.6	97.7	0.3
30	98.6	97.9	0.3

Table 4.7: Classification accuracy (%) for breast cancer dataset 2 using *RACAL-PS*, based on 100 experiments.

labelled objects %	Best (%)	Average (%)	Std (%)
5	95.8	90.6	4.7
10	96.3	92.1	3.2
15	96.8	93.5	2.3
20	97.5	94.4	1.8
25	97.9	94.9	1.6
30	98.3	95.2	1.7

Table 4.8: Classification accuracy (%) for leukaemia dataset using *RACAL-PS*, based on 100 experiments.

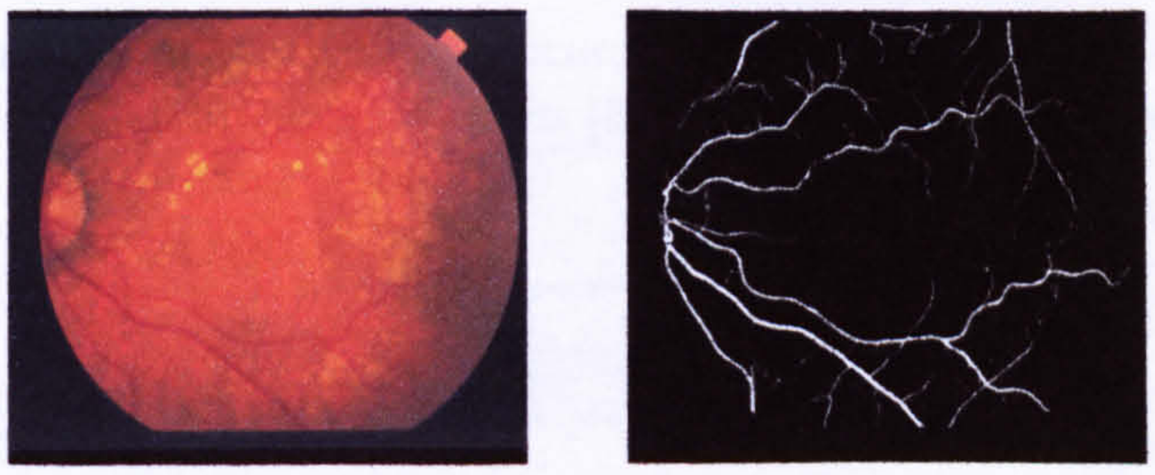
labelled objects %	Best (%)	Average (%)	Std (%)
10	94.4	80.2	10.6
20	97.2	89.6	4.2
30	97.2	90.9	3.0
40	97.2	93.1	2.4
50	100	94.7	1.8

Table 4.9: Classification accuracy (%) for iris dataset using *RACAL-PS*, based on 100 experiments.

labelled objects %	Best (%)	Average (%)	Std (%)
10	97.1	95.8	3.3
15	98.0	96.1	3.2
20	98.7	96.2	2.8
25	99.3	96.7	2.6
30	99.3	97.1	2.4

In these experiments, the performance of *RACAL-PS* is also examined in segmenting retinal blood vessels, where only 30% of all pixels are known as vessels or non-vessels pixels. Figure 4.14(a and b) shows two examples; abnormal (top) and normal (bottom) images and their results after blood vessels segmentation using *RACAL-PS*. In this application, the performance is measured with Receiver Operating Characteristic (ROC) curves [87]. An ROC curve plots the false positive rates against the true positive rates, and these rates are defined in the same way as in [88], where the true (false) positive is any pixel which was hand-labelled as a vessel (not vessel) and whose intensity after segmentation is above a given threshold. The true (false) positive rate is established by dividing the number of true (false) positives by the total number of pixels hand-labelled as vessels (not vessels).

By using the whole 20 retinal images, average sensitivity (true positive rate) of 81.34% is achieved at average specificity (1-false positive rate) of 96.70% as summarised in Table 4.10. These values are calculated using the retinal field of view only (FOV).



(a) (b)

Figure 4.14: (a) Colour images, and (b) output as hard decision using *RACAL-PS*.

Table 4.10: *RACAL-PS* hard decision results (average from 20 images).

Image type	Specificity %	Sensitivity %
Normal	97.02%	85.01%
Abnormal	96.39%	77.67%
All	96.70%	81.34%

4.6.2 Comparison with other classifiers

4.6.2.1 Breast cancer data

For purposes of comparison, the classification results obtained by *RACAL-PS* on breast cancer dataset 2 are compared with results from different classifiers [89] (*PCA/MDC* "Principal Component Analysis / Minimum Distance Classifier" [7, 11], *FLDA/MDC* "Fisher Linear Discriminant Analysis / MDC" [11], *MLP* "Multi-Layer Percepton" [90], *SVM* "Support Vector Machine" [91], and *GP/MDC* "Genetic Programming/ MDC" [89, 92]). In order to achieve fair comparisons as in [89], random samples (100 samples) are selected, from the entire dataset for training, and 100 samples for testing; this process has been repeated 100 times. The target information, class labels, of the training samples is used to guide

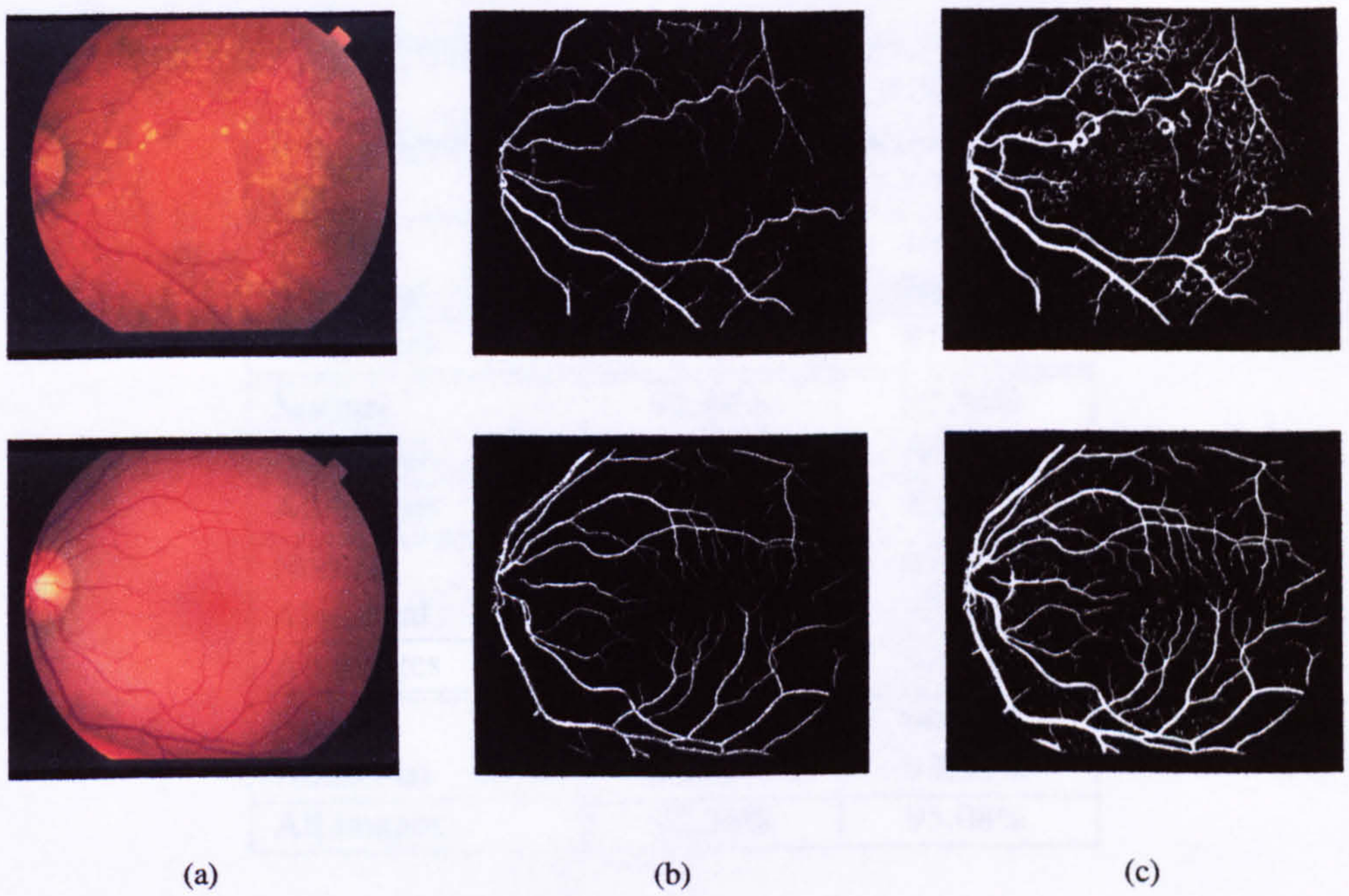
Table 4.11: Comparison of classification accuracy (%) for breast cancer dataset 2 (testing set) using *RACAL-PS* and different classifiers [89], based on 100 experiments.

Algorithms	Best (%)	Average (%)	Std (%)
PCA/MDC	88.7	88.6	N/A
FLDA/MDC	88.9	88.6	N/A
MLP	97.3	96.2	1.7
SVM	96.7	96.3	0.8
GP/MDC	98.9	97.4	1.5
RACAL-PS	100.0	97.3	1.6

the clustering process of the testing samples using *RACAL* algorithm. Table 4.11 shows the comparison results of *RACAL-PS* along with different methods for classification. As shown, the best classification accuracy is achieved by *RACAL-PS* (100%), with the lowest being 88.76% obtained by *PCA/MDC* which gives comparable results as *FLDA/MDC*. Although the average and standard deviation of classification accuracy obtained by *GP/MDC* is comparable with *RACAL-PS*, it gives 1.1% less than the best performance *RACAL-PS*. Therefore, the proposed method is more robust compared with the other methods.

4.6.2.2 Retinal images

For purposes of comparison, classification results obtained from the proposed algorithm (*RACAL-PS*) are compared with the classification results obtained by the *KNN* classifier. For the *KNN* classifiers, two sets are required; one for training and the other for testing, so the dataset is randomly divided into two sets of images, each contains 5 normal and 5 abnormal images. The training set contains large number of training samples (423,500 pixels/image), which is huge and is the main problem with this type of classifiers. To overcome such a problem, random number of pixels are chosen from the field of view (FOV) of each image in the training set [84]. The targets for these training samples are available from the manually segmented images. The testing set contains 10 images to test the performance of the classifier. While for *RACAL-PS* , only 30% of all pixels are known (as vessels or non-vessels pixels). Figure 4.15 compares results from *RACAL-PS* with the *KNN* classifier.



(a) (b) (c)

Figure 4.15: (a) Colour images, output as hard decision from (b) *RACAL-PS*, and (c) *KNN* classifier.

Table 4.12: *RACAL-PS* and *KNN* hard decision results (average from 10 images (testing set)).

Image type	*RACAL-PS* Specificity %	*RACAL-PS* Sensitivity %	*KNN* [84] Specificity %	*KNN* [84] Sensitivity %
Normal	97.18%	85.96%	93.56%	88.59%
Abnormal	96.90%	80.32%	91.92%	82.36%
All	97.04%	83.14%	92.74%	85.47%

For hard classification, the results are summarised in Table 4.12. As shown, average sensitivity of 83.14% is achieved at average specificity of 97.04% from *RACAL-PS* compared with average sensitivity of 85.47% at average specificity of 92.74% from the *KNN* classifier. On average, the proposed *RACAL-PS* achieves better specificity than *KNN* classifier with comparable sensitivity.

By comparing the soft classification results of *RACAL-PS* and *KNN* classifier as summarised in Table 4.13, it is clear that the proposed *RACAL-PS* offers better results than the *KNN* classifier for abnormal images, and comparable results for normal images.

For automation purposes, *RACAL-PS* can be used as a classifier by learning from 10

Table 4.13: Average sensitivity at certain specificity values.

	RACAL-PS	*KNN* [84]
Image type Specificity %	Sensitivity %	Sensitivity %
Normal	85.31%	86.60%
Abnormal 95%	81.04%	76.24%
All images	83.18%	81.42%
Normal	92.89%	92.56%
Abnormal 90%	93.53%	86.13%
All images	93.21%	89.35%
Normal	94.08%	95.03%
Abnormal 85%	97.70%	90.89%
All images	95.89%	92.96%
Normal	98.07%	96.51%
Abnormal 80%	96.64%	93.65%
All images	97.36%	95.08%

Table 4.14: *RACAL-PS* as a classifier and the *KNN* classifier hard decision results (average from 10 images (testing set)).

Image type	*RACAL-PS*		*KNN* [84]	
	Specificity %	Sensitivity %	Specificity %	Sensitivity %
Normal	98.64%	91.33%	93.56%	88.59%
Abnormal	98.32%	89.52%	91.92%	82.36%
All	98.48%	90.43%	92.74%	85.47%

images and testing on the other images. In the training step, each image is clustered to K clusters, then from ground truth images, each cluster is assigned to the corresponding class. Afterwards, describe each cluster statistically and geometrically by calculating mean of features for its objects, cluster compactness, major and minor diameters. For testing, cluster each image as in Section 4.2, then for each cluster; calculate the mean of features for its objects, cluster compactness, major and minor diameters. For each cluster in the testing image, find the nearest cluster "with known class" from the training set, then assign it to the same class. Results to compare between *RACAL-PS* as a classifier and the *KNN* classifier are shown in Table 4.14. As shown, average sensitivity of 90.43% is achieved at average specificity of 98.48% from *RACAL-PS* compared with average sensitivity of 85.47% at average specificity of 92.74% from the *KNN* classifier. On average, the proposed *RACAL-PS* performs better than the *KNN* classifier.

4.7 Parallel Implementation

In this section, a parallel version of *RACAL* clustering algorithm is proposed to cluster large datasets with high dimensions. The proposed algorithm (*P-RACAL*) is based on the Single Program Multiple Data (SPMD) model using message passing. A group of massively parallel processor machines or network of PCs can be used by the proposed parallel *RACAL* (*P-RACAL*). The problem of large datasets is addressed by partitioning the dataset into small subsets. These subsets are distributed among processors. i.e, each processor has its own local data (subset). The communication between processors is managed by MPI, the Message Passing Interface, which is a standard and portable message-passing system [93]. The proposed algorithm (*P-RACAL*) is detailed in Algorithm 4. The speedup and scaleup measurements are used to measure the performance of parallel algorithms [46, 48]. The speedup captures the gain obtained when an algorithm is run in parallel and, ideally should equal to the number of processors. In other words, speedup is defined as the ratio between the execution time when using one processor to the execution time when using p processors as described in Eq. 4.3. The scaleup captures how a parallel algorithm handles larger datasets where more processors are available. It can be achieved by fixing the problem size per processor while increasing the number of processors.

$$Speedup = \frac{T_S}{T_P} \tag{4.3}$$

where T_S is the sequential execution time, and T_P is the parallel execution time.

To examine the performance of *P-RACAL* in parallel environments, *P-RACAL* is implemented on a computer cluster using the standard MPI which allows *P-RACAL* algorithm to be completely portable to the other shared-nothing parallel architectures. All experiments are carried out on a cluster of 12 nodes, each node consists of 4 processors (each, Xeon 2.8 GHZ CPU with 512 MB RAM). These nodes are connected via a fast Ethernet network.

In this study, retinal images are used to examine the effect of data size n and the number of features d on the speedup and scaleup of *P-RACAL*, where $n = 423,500$ which corre-

Algorithm 4 The Proposed Parallel Algorithm *P-RACAL*.

Read the input data $O = \{O_1, \ldots, O_n\}$, the radius value δ_o, the number of processors p.
Compute the partition (block) length per processor, n_p:

$$n_p = \begin{cases} n/p & if \quad p\ is\ even \\ n/p + 1 & if \quad p\ is\ odd \end{cases}$$

where n is the data size.
Each processor:

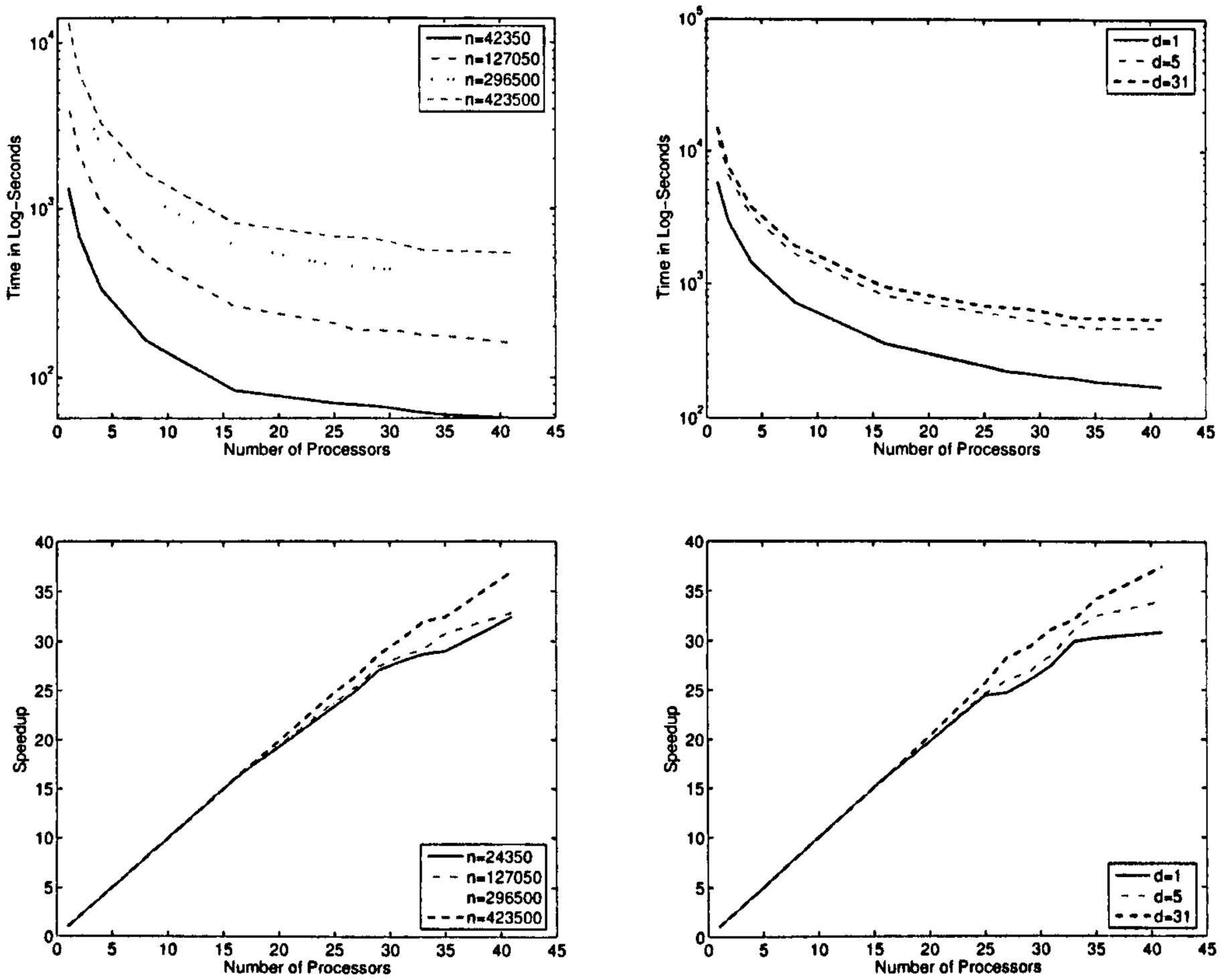
1. loads its local data subset (partition).
2. finds the neighbourhood and the weight of each object in its local data according to the input radius δ_o.
3. creates a list that contains neighbourhoods of some objects from other processors.
4. communicates with other processors in such a way that every two processors can communicate at a time. It should be noted that there will be one processor in idle case when p is odd.

Apply the client-server process strategy:
Let P_{server} the server processor and P_{client} a client processor.

1. *Server*: collects the weights from all clients (other processors).
2. **Generating prototypes (cluster seeds)**
   Do
   Server: finds the max weighted object O_{max}, clears the weights of its neighbourhood (closest objects), and sends a request to the client that have O_{max} asking for the common objects between O_{max} and its neighbourhood.
   Clients: if P_{client} is the requested processor, then send to P_{server} the common objects between O_{max} and its neighbourhood, else P_{client} remains idle until receiving permission from P_{server}.
   Server: calculates mean of the common objects between O_{max} and its neighbourhood to generate a prototype B_k (cluster seed).
   Until no weights are found
3. **Clustering process**
   Do
   Server: broadcasts the prototypes (cluster seeds) to all clients.
   Server: assigns each object O_i - in its local data - to the nearest prototypes, and requests the local partition from all clients.
   Clients: assign each object O_i - in its local data - to the nearest prototypes. Then send its local partition to P_{server} and remain idle until receiving permission from P_{server}.
   Server: calculates the validation index value as in Eq. 3.3.
   Server: collects the local partitions and then update the prototypes (cluster centres).
   Until no more changes
4. *Server*: saves the clustering result of the best validation value.

sponds to a retinal image of 605 × 700 pixels with 31 features per pixel [94]. The effect of data size n is obtained by varying n and fixing the number of features d, i.e, one can get different dataset corresponding to each variation of n, and then, apply *P-RACAL* on each dataset several times using different number of processors. While, the effect of number of features d is obtained by varying d and fixing the data size n, i.e, one can get different dataset corresponding to each variation of d, then, apply *P-RACAL* on each dataset several times using different number of processors.

Results from varying the data size n and the number of features d are plotted in Fig. 4.16. Figure 4.16(a) shows the effect of varying the data size n on the execution time. For small data sizes, the execution time is reduced faster than that for larger data sizes, which is consistence with the obtained speedup. On the other hand, a large data size gives nearly linear speedup at high number of processors. Similarly, when varying the number of features d and fixing the data size n as shown in Fig. 4.16(b), the observed speedup for larger number of features remains nearly linear at high number of processors. Consequently, the proposed *P-RACAL* is not affected by the number of features which indicate the scalability. It should be noted that the above results are affected by the communication overhead among processors. Figure 4.17 shows the relative scaleup results when the number of processors and the data size are simultaneously increased. Where, the relative scaleup of *P-RACAL* is defined as the ratio between the execution time for clustering n pixels using one processor to the execution time for clustering $n \times p$ pixels using p processors. As shown, *P-RACAL* delivers nearly constant execution time. It should be noted that the convergence time when running the algorithm using single processor is much higher than its convergence time when it runs in parallel. However, it is noteworthy to take into consideration the effect of problem sizes. For small problem sizes, the communication overhead among processors has a significant effect on the convergence time compared with large problem sizes.



(a) (b)

Figure 4.16: (a) The execution time versus number of processors and its corresponding speedup when using four different data sizes n at $d = 31$, and (b) the execution time versus number of processors and its corresponding speedup when using three different dimensions d at $n = 423,500$.

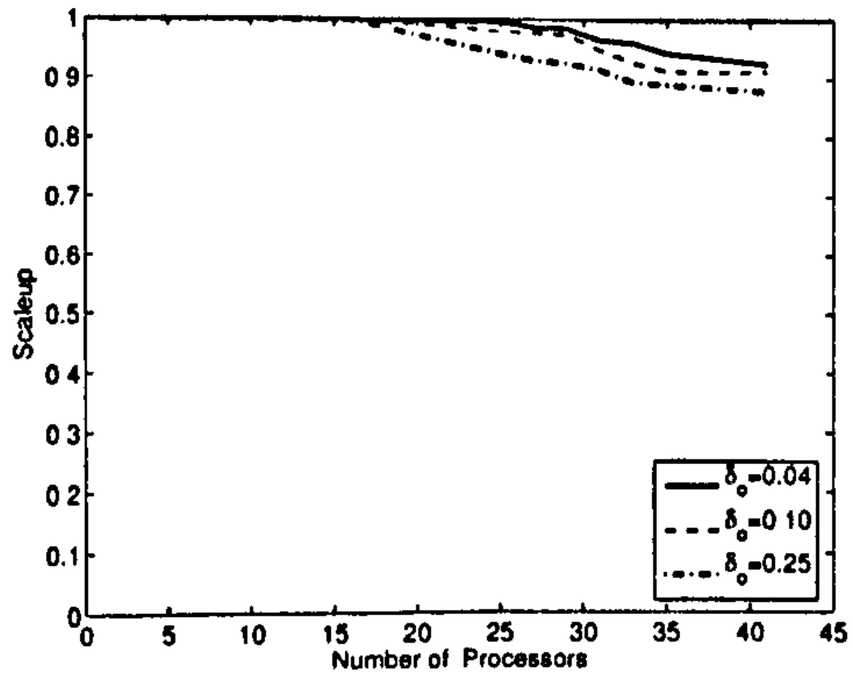


Figure 4.17: Scaleup versus number of processors at different δ_o values.

4.8 Summary

In this chapter, a new clustering algorithm, *RACAL*, has been presented. As shown, it is possible to control the behaviour of the cluster membership (small or large) through a radius parameter δ_o. Unlike many partitional clustering algorithms, which require the specification of the number of required clusters, the proposed *RACAL* does not need a priori knowledge of the number of clusters. Moreover, the augmentation of *RACAL* with a reliable validation index produces the best results for the given δ_o values. Four different types of real-world data have been used to demonstrate that the proposed algorithm scales well with the size and dimension of the dataset. The clustering performance measures, *CPMs*, indicate that *RACAL* possesses higher robustness as well as offers better repeatability compared with other mentioned clustering algorithms. Moreover, an adaptive partial supervision (*PS*) strategy has been proposed for *RACAL* to make it act as a classifier. Experimental classification results obtained by *RACAL-PS* indicate its superior performance compared with the other classification methods. Additionally, a parallel version of *RACAL* (*P-RACAL*) has been proposed. As demonstrated, *P-RACAL* is scalable in terms of speedup and scaleup, which gives the ability to handle large datasets of high dimensions in a reasonable time.

Chapter 5

Nearest Neighbour Clustering Algorithm (NNCA)

5.1 Introduction

IN the previous chapter, a radius based clustering algorithm (*RACAL*) has been proposed. This algorithm, *RACAL*, achieves clustering using a radius parameter δ_0 which acts as the determinant of the cluster and controls the size of the clusters, which is the main constraint of *RACAL* algorithm.

In this chapter, a novel clustering algorithm which achieves clustering without any control of cluster sizes is proposed. The proposed clustering algorithm, which is called Nearest Neighbour Clustering Algorithm (*NNCA*), uses the same concept as the K-nearest neighbour (*KNN*) [84] classifier with the advantage that the algorithm needs no training set and it is completely unsupervised. Additionally, a partial supervision strategy is proposed for using in conjunction with *NNCA* to make it act as a classifier. Furthermore, a parallel algorithm for *NNCA* is proposed to reduce the clustering time of large datasets.

5.2 The NNCA Clustering Algorithm

NNCA is a modified version of the *KNN* classifier which requires human intervention to define a training set, i.e. supervised, while *NNCA* needs no training set and it is completely unsupervised. In this context, *NNCA* is divided into two stages for creating N_C clusters. The first stage is to randomly select N objects (the choice of N should be large enough to describe the distribution of the data), where an object is a single data item used by the clustering algorithm and is characterised by d features (attributes). Then non-overlapping clusters are created from these N objects, each of maximum size K_{init} objects (the choice of K_{init} ensures that more than N_C clusters are generated here). Afterwards an iterative control strategy is applied to update the clusters and their memberships by increasing the number of neighbours until N_C non-overlapping clusters are created. The second stage is to cluster the remaining objects. For each unclustered object q, K nearest clustered objects are found. Then, the cluster to which most of these K clustered objects belong is deemed to be one to which the object q belongs to.

The *NNCA* clustering algorithm is detailed in Algorithm 5. Let each object x be described by the feature vector:

$$< a_1(x)\, a_2(x)\, , \ldots\ldots\, , a_d(x) >$$

where $a_r(x)$ is used to denote the values of the r-th attribute of data object x. If two objects x_i and x_j are considered, then the distance between them is defined as $D(x_i, x_j)$, which is expressed in Eq. 5.1.

$$D(x_i, x_j) = \sqrt{\sum_{r=1}^{d} (a_r(x_i) - a_r(x_j))^2} \tag{5.1}$$

A fuzzy clustering, where all objects are allowed to belong to all clusters with different degrees of membership, is achieved by obtaining the mean value of the K nearest neighbours for each object in the dataset. Therefore, hard partition as well as soft partition can be obtained. For an object x_q to be clustered, let $x_1, \ldots, x_K$ denote the nearest K clustered objects to x_q and $C(x_i) \in \{1, \ldots, N_C\}$ be the cluster index for object x_i.

Algorithm 5 Nearest Neighbour Clustering Algorithm (*NNCA*)

Input (data, N, K_{init}, N_C, K)

where:

* N is the number of random objects to be clustered.
* K_{init} is the nearest neighbour objects from N.
* N_C is the user defined number of clusters.
* K is the number of nearest clustered objects.

Step 1: Create N_C non-overlapped clusters

(a) Create initial clusters:

* Initially, all the N objects are unclustered.

let $M = 1$

For $i = 1$ to N

IF (object i is unclustered)

- Assign i and its unclustered neighbours (from N) of the K_{init} nearest neighbours to cluster # M.
- $M = M + 1$

End IF

End For

(b) Merge clusters:

* **DO**

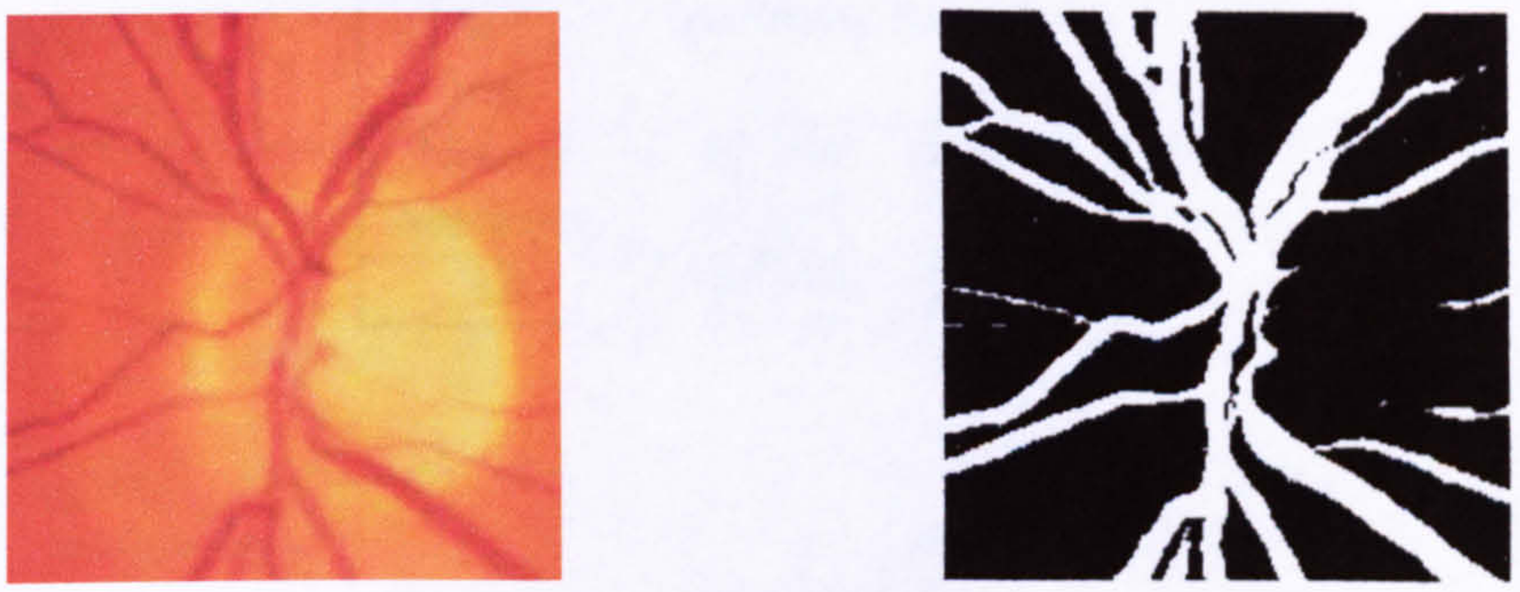
- $K_{init} = K_{init} + 1$
- Assign each clustered object to the common cluster of the K_{init} nearest neighbours.
- Update the number of clusters $\rightarrow M$

WHILE ($M > N_C$)

Step 2: Find the nearest K neighbours for each remaining object

- Assign each unclustered object to the common cluster of the K nearest clustered objects.
-Use Eq. 5.2 to find hard partition and Eq. 5.3 to find soft partition.

Output (Hard partition vector, Soft partition matrix)



(a) (b)

Figure 5.1: (a) Original sub-image, and (b) sub-image with blood vessels clustered using *NNCA*.

The hard partition for x_q is:

$$C(x_q) = \arg\max_{n \in N_C} \sum_{r=1}^{K} (n == C(x_r)), \tag{5.2}$$

and soft partition is:

$$C(x_q) = \frac{\sum_{r=1}^{K} C(x_r)}{K} \tag{5.3}$$

Figure 5.1 shows the result for clustering blood vessels for a sub-image from a colour retinal image.

When using 20 images where each pixel is characterised by three features [84], average sensitivity of 77% is achieved at average specificity of 90% as summarised in Table 5.1. These values are calculated using the retinal field of view only. The setting parameters of *NNCA* were $N = 50,000$, $K_{init} = 200$, $N_C = 2$, and $K = 60$.

Table 5.1: *NNCA* results (average from 20 images).

Image type	Specificity %	Sensitivity %
Normal	92.30%	81.42%
Abnormal	87.61%	72.13%
All	89.95%	76.77%

5.3 NNCA with Partial Supervision Strategy (NNCA-PS)

In this section, adaptation of *NNCA* algorithm with some labelled objects is proposed to guide the clustering process of the unlabelled objects, i.e., *NNCA* with partial supervision (*NNCA-PS*). The proposed method is divided into two stages. First stage is to randomly select N_P objects from the dataset to be labelled data objects and cluster these N_P objects into N_C clusters, as described in Section 5.2. Second stage is to classify each cluster according to the class label of the majority of its objects. For each labelled data object x_l of class C_i, assigned to cluster j ($1 \leq j \leq N_C$), if its cluster is classified to different class (label), then this data object will be assigned to the cluster that has the nearest objects and with the same label of it as in Eq. 5.4.

$$j = \begin{cases} j & if\ cluster\ j\ \in C_i \\ \arg\min\limits_{k \in C_i} \dfrac{\sum_{z \in k} D_{z x_l}}{|cluster\ k|} & if\ cluster\ j\ \notin C_i \end{cases} \tag{5.4}$$

where $|cluster\ k|$ is the number of objects in cluster k, and $D_{z x_l}$ is the euclidean distance between an object z and the labelled object x_l.

This process continues until all labelled objects within a cluster have the same class label. Then, the process continues to assign each unlabelled object x_u to the cluster that has the nearest labelled objects as in Eq. 5.5. Then, all the data objects that belong to different clusters with the same class labels can be assigned to that label.

$$j = \arg \min_{1 \leq k \leq N_C} \frac{\sum_{z \in k} D_{z x_u}}{|cluster\ k|} \tag{5.5}$$

where $D_{z x_u}$ is the euclidean distance between an object z and the unlabelled object x_u.

Algorithm 6 *NNCA* with partial supervision strategy (*NNCA-PS*)

- *Step 1: Clustering using NNCA algorithm*

  1. Randomly select N_P points from the ground truth to be labelled objects.
  2. Cluster the N_P objects into N_C cluster using *NNCA* clustering algorithm.

- *Step 2: Apply the supervision strategy as follow:*

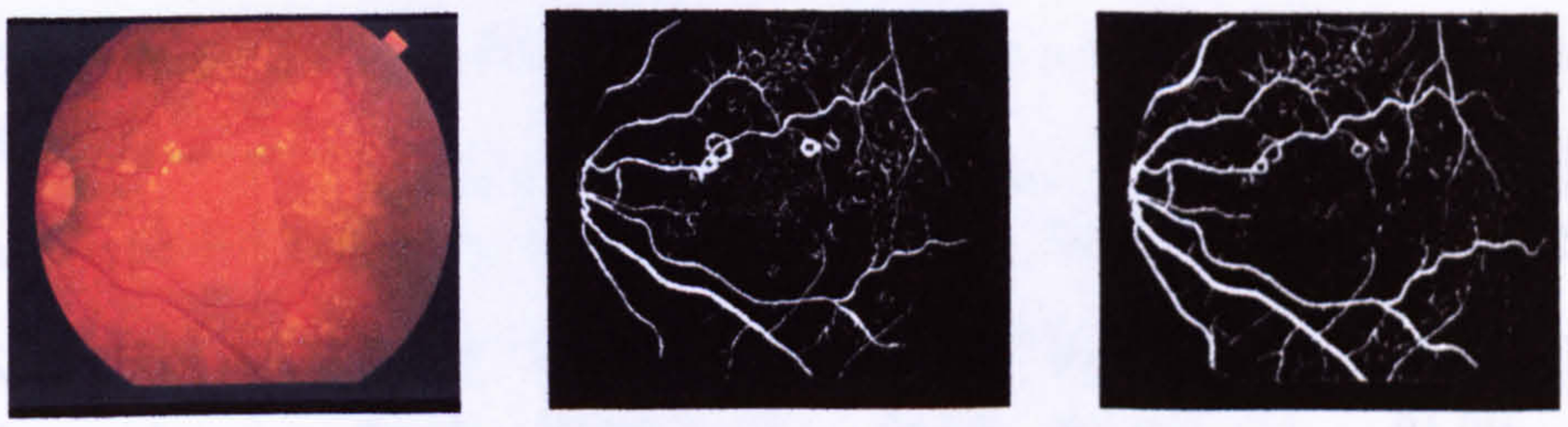
  1. Classify the clusters obtained by *NNCA* algorithm to the class of its most labelled objects.
  2. For each labelled object, if its cluster is classified to different class (label), then this object will be assigned to the cluster that has the nearest objects and with the same label of it.
  3. Each unlabelled object is assigned to the cluster that has the nearest objects and then classified to the class (label) of this cluster.

The proposed supervised method is detailed in Algorithm 6. In order to reduce the misclustered objects and finally achieve higher classification performance, the average separation between all possible pairs of clusters (cluster centres) is calculated at each update of the neighbourhoods in step 1(b) of Algorithm 5. Then, the clustering result that has maximum separations between clusters is used in conjunction with the proposed partial supervision strategy (*NNCA-PS*) to achieve better classification accuracy.

The soft classification of *NNCA-PS* is achieved in the same way as the soft classification of *RACAL-PS* (see Eq. 4.1 and 4.2), where each objects is allowed in principle to belong to all classes with different degrees of membership, achieved by adding the fuzzy memberships for each object with the clusters that belong to the same class label.

5.4 Experimental Results

The performance of the proposed algorithm using datasets from real-world problems is examined. The proposed method is used to segment the blood vessels from retinal images, and it is used to classify breast tumors as either malignant or benign.



(a) (b) (c)

Figure 5.2: (a) Colour images, output as hard decision from (b) *NNCA*, and (c) *NNCA-PS*.

Table 5.2: *NNCA* and *NNCA-PS* results (average from 10 images).

Image type	*NNCA* Specificity %	*NNCA* Sensitivity %	*NNCA-PS* Specificity %	*NNCA-PS* Sensitivity %
Normal	91.7%	83.4%	95.4%	90.2%
Abnormal	87.2%	77.7%	94.4%	87.8%
All	89.5%	80.5%	94.8%	89.0%

5.4.1 Retinal images

In these experiments, retinal blood vessels are segmented using the *NNCA* and *NNCA* with partial supervised strategy (*NNCA-PS*). For *NNCA-PS*, only 30% of all the pixels are known as vessels or non-vessels pixels. Figure 5.2(a and b) shows two examples; abnormal (top) and normal (bottom) images and their results after blood vessels segmentation using *NNCA* and *NNCA-PS*. As shown in Table 5.2, *NNCA-PS* achieves average sensitivity (true positive rate) of 89% at average specificity (1-false positive rate) of 94.8%, while the *NNCA* achieves sensitivity of 80.5% at average specificity of 89.5%. On average, the *NNCA-PS* achieves better specificity as well as sensitivity than *NNCA* as it is completely unsupervised. This indicates the effect of labelled objects in improving the clustering process.

Table 5.3: *NNCA-PS*, *RACAL-PS* and *KNN* hard decision results (average from 10 images (testing set)).

Image type	*NNCA-PS*		*RACAL-PS*		*KNN* [84]	
	Specificity %	Sensitivity %	Specificity %	Sensitivity %	Specificity %	Sensitivity %
Normal	95.4%	90.2%	97.2%	85.9%	93.6%	88.6%
Abnormal	94.4%	87.8%	96.9%	80.3%	91.9%	82.4%
All	94.8%	89.0%	97.0%	83.1%	92.7%	85.5%

Table 5.4: Average sensitivity at certain specificity values for 10 images.

	NNCA-PS	*RACAL-PS* [95]	*KNN* [84]
Image type Specificity %	Sensitivity %	Sensitivity %	Sensitivity %
Normal	90.8%	85.3%	86.6%
Abnormal 95%	86.7%	81.0%	76.2%
All images	88.8%	83.2%	81.4%
Normal	95.1%	92.9%	92.6%
Abnormal 90%	92.8%	93.5%	86.1%
All images	93.9%	93.2%	89.4%
Normal	96.9%	94.1%	95.1%
Abnormal 85%	95.4%	97.7%	90.9%
All images	96.1%	95.9%	92.9%
Normal	98.1%	98.1%	96.5%
Abnormal 80%	96.9%	96.6%	93.7%
All images	97.5%	97.4%	95.1%

For purposes of comparison, the performance of *NNCA-PS* is also compared with *KNN* classifier [84] and *RACAL-PS* [95]. For hard classification in *NNCA-PS*, the same set of images (10 images for testing) as used with the *KNN* classifier are studied. The hard classification results from *NNCA-PS*, *RACAL-PS* and *KNN* are presented in Table 5.3. As shown, *NNCA-PS* achieves average sensitivity of 89% at average specificity of 94.8%, while the *KNN* classifier achieves sensitivity of 85.5% at average specificity of 92.7%. On average, the proposed *NNCA-PS* achieves better specificity as well as sensitivity than *KNN* classifier. When comparing with *RACAL-PS*, although *RACAL-PS* achieves 2% higher specificity (on average) than *NNCA-PS*, it gives 6% less sensitivity (on average) than *NNCA-PS*.

For soft classification as shown in Table 5.4, the soft classification results of the proposed *NNCA-PS* are compared with the soft results of *RACAL-PS* and *KNN*. As shown, at 95% specificity, the proposed *NNCA-PS* achieves 5.5% and 4.2% higher sensitivity than

RACAL-PS and *KNN* respectively in case of normal images. Also in abnormal images at 95% specificity, *NNCA-PS* achieves 5.7% and 10.5% higher sensitivity than *RACAL-PS* and *KNN* respectively. For higher specificity, *KNN* classifier achieves the lowest average sensitivity compared with *NNCA-PS* and *RACAL-PS*, while both *NNCA-PS* and *RACAL-PS* achieves on average a comparable sensitivity.

5.4.2 Breast cancer data

For purposes of comparison, a series of experiments were carried out to examine the performance of *NNCA* when applying the proposed supervision strategy (*NNCA-PS*) on breast cancer datasets, where the classification results obtained by *NNCA-PS* on breast cancer dataset 2 are compared with results of different classifiers [89]. As shown in Table 5.5, the best classification accuracy is achieved by *NNCA-PS* (99.5%), with the lowest being 88.7% obtained by *PCA/MDC* which gives comparable results as *FLDA/MDC*. Although the average classification accuracy obtained by *GP/MDC* is comparable with *NNCA-PS*, it gives 0.6% less than the best performance of *NNCA-PS* with higher standard deviation in classification accuracy. Therefore, the proposed method is more robust compared with other methods.

Table 5.5: Comparison of classification accuracy (%) for breast cancer dataset 2 (testing set) using *NNCA-PS* and different classifiers [89], based on 100 experiments.

Algorithms	Best (%)	Average (%)	Std (%)
PCA/MDC	88.7	88.6	N/A
FLDA/MDC	88.9	88.6	N/A
MLP	97.3	96.2	1.7
SVM	96.7	96.3	0.8
GP/MDC	98.9	97.4	1.5
NNCA-PS	99.5	97.2	1.2

In order to reduce the amount of a priori knowledge, a small number of objects from the entire dataset are used as labelled objects. In these experiments, the effect of the number of labelled objects on the classification accuracy are investigated. Fractions of objects from the entire dataset are randomly selected to be used as labelled objects. For each fraction, this process is repeated one hundred times. The best, average, and standard deviation

Table 5.6: Classification accuracy (%) for breast cancer dataset 2 (entire dataset) using *NNCA-PS*, based on 100 experiments.

labelled objects %	Best (%)	Average (%)	Std (%)
5	96.2	91.5	2.3
10	96.3	93.1	1.8
15	97.0	94.4	1.3
20	97.2	95.3	1.0
25	97.6	95.6	0.9
30	98.2	96.3	0.7

Table 5.7: Classification accuracy (%) for breast cancer dataset 1 (entire dataset) using *NNCA-PS*, based on 100 experiments.

labelled objects %	Best (%)	Average (%)	Std (%)
5	98.0	96.0	1.2
10	98.1	96.3	1.1
15	98.5	96.7	0.9
20	98.7	97.0	0.8
25	98.7	97.4	0.7
30	99.2	97.9	0.5

of classification accuracy are obtained over one hundred runs for each fraction of labelled objects. For breast cancer dataset 2 as demonstrated in Table 5.6, the best and average classification accuracies increase with the increasing fraction of the labelled objects. As shown, the best and average classification accuracy of 98.2% and 96.3% respectively were achieved at 30% labelled objects, with the lowest being 96.2% and 91.5% for best and average accuracies respectively at 5% labelled objects. By examining the average and standard deviation of the classification performance, when 5% of the entire dataset are labelled, the average performance is the lowest, while it has the highest standard deviation compared with other fractions of labelled objects. For breast cancer dataset 1 as demonstrated in Table 5.7, standard deviations are lower than standard deviations of breast cancer dataset 2. This is because the higher compactness of clusters on breast cancer dataset 1 compared with breast cancer dataset 2 [86]. For 5% labelled objects and higher, the best classification accuracy is higher than 98% with a small decrease in the standard deviation and a significant increase in the average classification accuracy as in Table 5.7.

When comparing the proposed *NNCA-PS* with *RACAL-PS* for breast cancer data classification, where a small number of objects from the entire dataset are used as labelled objects. The average classification accuracy for breast cancer dataset 2 using *NNCA-PS* is 1% higher than *RACAL-PS* while it achieves comparable accuracy for breast cancer dataset 1 as demonstrated in Tables 5.8 and 5.9. Moreover, the standard deviation of the classification performance of *NNCA-PS* for breast cancer dataset 2 is lower than *RACAL-PS* which favors compact clusters, while it achieves slightly higher standard deviations in breast cancer dataset 1. This may be the result of achieving clustering without any control of cluster sizes in *NNCA-PS*, while *RACAL-PS* is constrained with a radius parameter δ_0 which controls the size of the clusters.

Table 5.8: Comparison of classification accuracy (%) for breast cancer dataset 2 (entire dataset) using *NNCA-PS* and *RACAL-PS*, based on 100 experiments.

labelled objects %	*NNCA-PS* Average(%) ± Std(%)	*RACAL-PS* Average(%) ± Std(%)
5	91.5 ± 2.3	90.6 ± 4.7
10	93.1 ± 1.8	92.1 ± 3.2
15	94.4 ± 1.3	93.5 ± 2.3
20	95.3 ± 1.0	94.4 ± 1.8
25	95.6 ± 0.9	94.9 ± 1.6
30	96.3 ± 0.7	95.2 ± 1.7

Table 5.9: Comparison of classification accuracy (%) for breast cancer dataset 1 (entire dataset) using *NNCA-PS* and *RACAL-PS*, based on 100 experiments.

labelled objects %	*NNCA-PS* Average(%) ± Std(%)	*RACAL-PS* Average(%) ± Std(%)
5	98.0 ± 1.2	97.5 ± 1.4
10	98.1 ± 1.1	97.9 ± 0.3
15	98.5 ± 0.9	98.2 ± 0.3
20	98.7 ± 0.8	98.6 ± 0.3
25	98.7 ± 0.7	98.6 ± 0.3
30	99.2 ± 0.5	98.6 ± 0.3

5.5 Parallel Implementation

In this section, a parallel version of *NNCA* algorithm is proposed. The proposed *P-NNCA*, Parallel *NNCA*, algorithm is based on the Single Program Multiple Data (SPMD) model using message passing. The problem of large datasets is addressed by partitioning the dataset into small subsets. These subsets are distributed among processors. i.e, each processor has its own local data (subset). The proposed *P-NNCA* is detailed in Algorithm 7.

5.5.1 Fast K nearest neighbours process

The problem of finding the nearest K neighbours for a data object (pixel) x_i can be achieved by either sorting algorithms or recursive methods. There is no doubts that these methods suffer from large computational complexity specially for higher number of data objects. In this study, a new method for obtaining the nearest K neighbours for a data object is proposed, which is called Fast K Nearest Neighbours (*FKNN*). This method aims to minimise the number of comparisons which are necessary to find the nearest neighbours that correspond to the smallest distances (highest similarities). The proposed method is based on breaking the distance vector of length n (data size) for a data object x_i into small subsets (vectors) of length l. For each subset, find the smallest distance, i.e, one can get $\frac{n}{l}$ distances. The first nearest neighbour is the one that corresponds to the smallest distance from the $\frac{n}{l}$ distances. Then, find the second smallest distance within this subset, that contains the first nearest distance, and compare between it and the smallest distances from other subsets ($(\frac{n}{l} - 1)$ distances previously found). This means that only the subset that has the smallest distance will look for the next smallest distance, i.e, instead of searching through the entire distance vector (of length n), search through a smaller subset (of length l). This process will continue until the nearest K neighbours are identified.

Now, it would be worth the effort to find the optimum subset length l to minimise the time necessary to find the nearest K neighbours. Let:

- D is a distance vector of length n (data size)
- K is the number of neighbours

Algorithm 7 The Proposed Parallel Algorithm *P-NNCA*.

- Read the input parameters of *NNCA* algorithm and the number of processors p.
- Compute the partition (block) length per processor, n_p:

$$n_p = \begin{cases} n/p & if \quad p\ is\ even \\ n/p+1 & if \quad p\ is\ odd \end{cases}$$

  where n is the data size.
- Each processor loads it local data subset (partition).

Apply the client-server process strategy:
Step 1: Create N_C non-overlapped clusters
(a) Create initial clusters:

1. *Server*: creates a list l_N of randomly selected N objects to be clustered.
2. *Server*: broadcasts the l_N list to all clients, and finds the nearest K_{init} neighbours for each object in the l_N if it is belong to its local data.
3. *Clients* : receive the l_N list, and finds the nearest K_{init} neighbours for each object in the l_N list if it is belong to its local data.
4. *Server*: collects the neigbourhoods from clients. and creates non-overlapped clusters as demonstrated in Algorithm 5.

(b) Merge clusters:

1. *Server*: increments the neighbourhoods, i.e, $K_{init} = K_{init} + 1$.
2. *Server*: broadcasts the neighbourhoods K_{init} and the clusters' memberships to all clients, and then assigns each clustered object in its local data to the common cluster of the K_{init} nearest neighbours (from N).
3. *Clients*: receive the neighbourhoods K_{init} and the clusters' memberships, and assign each clustered object in its local data to the common cluster of the K_{init} nearest neighbours (from N).
4. *Server*: updates the clusters' memberships. if the updated number of clusters (M) is greater than the desired number of clusters (N_c), then go to step (b-1).

Step 2: Find the K neighbours for each remaining object

1. *Server*: broadcasts the updated clusters' memberships to all clients, and assigns each unclustered objects in its local data to the common cluster of the K nearest clustered objects.
2. *Clients*: receive the updated clusters' memberships, and assign each unclustered objects in its local data to the common cluster of the K nearest clustered objects.
3. *Server*: collects the updated clusters' memberships, and save the results.

- C_K is the number of comparisons to find the nearest K neighbours,
- l is the subsets length.

The number of comparisons per subset = l, and the number of comparisons to find the smallest distance out of l distances = number of subsets = $\frac{n}{l}$.

For $K = 1$: There is no need to divide D into small subsets. Therefore, the minimum number of comparisons to find the nearest neighbour is n, Therefore, $C_1 = n$.

For $K = 2$: The distance vector D can be divided into small subsets. Then, C_2 = The number of comparisons to find the first minimum distance + The number of comparisons to find the second minimum distance, i.e, $C_2 = (l \times \frac{n}{l} + \frac{n}{l}) + (l + \frac{n}{l})$

For $K = 3$: $C_3 = C_2 + (l + \frac{n}{l})$

For $K = 4$: $C_4 = C_3 + (l + \frac{n}{l})$

Therefore, for $K = k$:

$$C_k = C_{k-1} + (l + \frac{n}{l})$$

or

$$C_k = (l \times \frac{n}{l} + \frac{n}{l}) + (l + \frac{n}{l}) \times (k-1) = n + (l \times k) - l + \frac{n}{l} \times k$$

The optimum subset length l can be obtained as:

$$\frac{dC_k}{dl} = k - 1 - \frac{n}{l^2} \times k = 0$$

Therefore, the optimum subset length l for $k \geq 2$:

$$l = \sqrt{n \times (\frac{k}{k-1})} \tag{5.6}$$

Figure 5.3 shows the execution time of different procedures to find the nearest K neighbours that correspond to the minimum K distances out of 423,500 distances (distances between a pixel x_i and 423,500 pixels for a retinal image of size 605×700 pixels). As shown, the proposed method (*FKNN*) gives lower execution time at all different values of K compared with bubble and selection sorting methods, and the partial sorting method (selection sorting for only K steps).

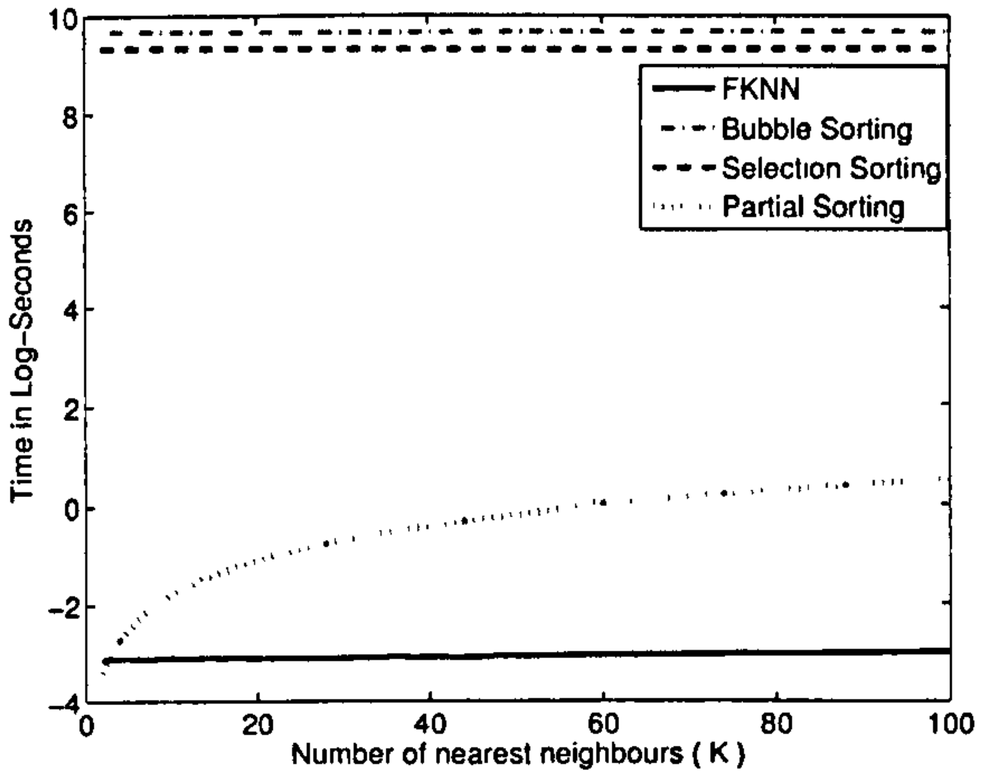


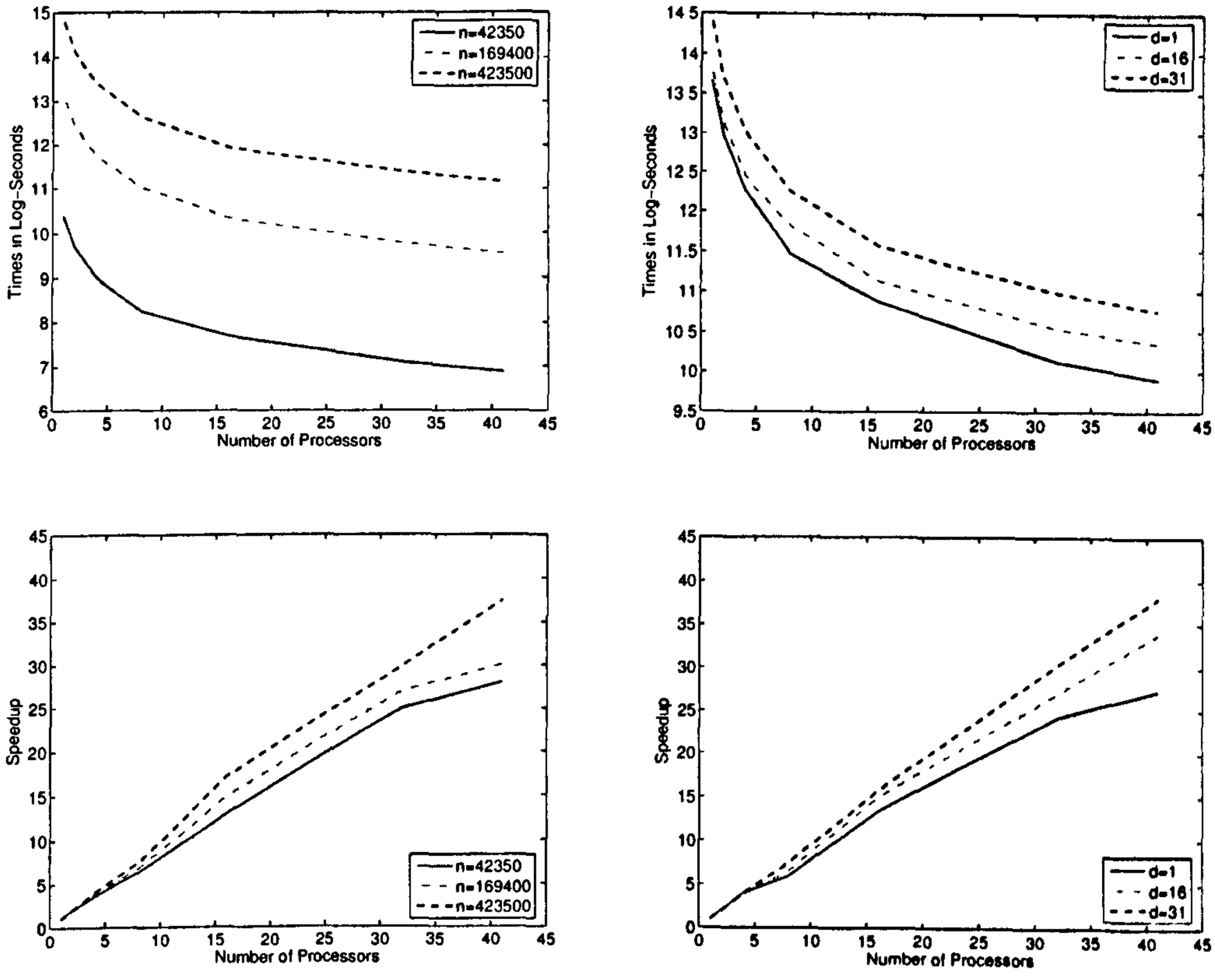
Figure 5.3: Execution times of different procedures to find the nearest K neighbours.

5.5.2 Parallel evaluation

To examine the performance of *P-NNCA* in parallel environments, *P-NNCA* is implemented on a computer cluster (as in Section 4.7) using the standard MPI which allows *P-NNCA* to be completely portable to the other shared-nothing parallel architectures.

The speedup and scaleup of *P-NNCA* are examined in the same way the speedup and scaleup of *P-RACAL* (see Section. 4.7). Results from varying the data size n and the number of features d are plotted in Fig. 5.4. Figure 5.4(a) shows the effect of varying the data size n on the execution time, while Fig. 5.4(b) shows the effect of varying the number of features d and fixing the data size n. As shown, the observed speedup for larger data sizes and larger number of features remains nearly linear at high number of processors. This indicate the scalability of the proposed *P-NNCA*. Figure 5.5 shows the relative scaleup results when the number of processors and the data size are simultaneously increased. As shown, *P-NNCA* delivers nearly constant execution time.

By comparing the parallel performances of *P-NNCA* and *P-RACAL* algorithms, *P-RACAL* achieves more linear speedup than *P-NNCA* as it requires more processing power in generating the prototypes (cluster centres).



(a) (b)

Figure 5.4: (a) The execution time versus number of processors and its corresponding speedup when using three different data sizes n at $d = 31$, and (b) the execution time versus number of processors and its corresponding speedup when using three different dimensions d at $n = 423,500$.

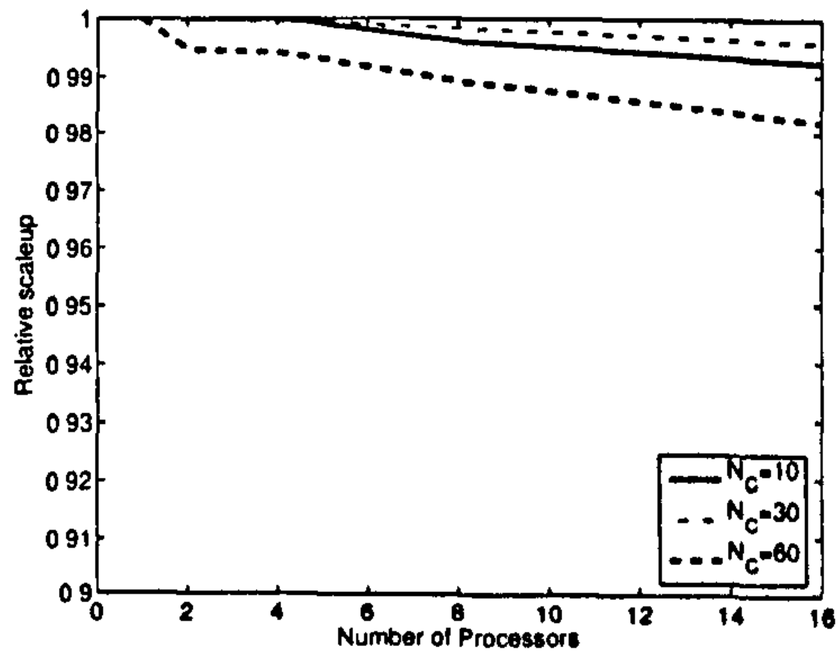


Figure 5.5: Scaleup versus number of processors at different number of clusters (N_C).

5.6 Discussion

When applying the proposed *NNCA-PS* algorithm for segmenting blood vessels from retinal images, *NNCA-PS* gives promising results of 89% average sensitivity at average specificity of 94.8% for hard classification. Furthermore, the results are compared with *KNN* classifier [84] and *RACAL-PS* [95] as shown in Table 5.3, where *KNN* classifier achieves 85.5% sensitivity at 92.7% specificity while *RACAL-PS* achieves 83.1% sensitivity at 97% specificity. On average, the proposed *NNCA-PS* achieves better specificity as well as sensitivity than *KNN* classifier. When comparing with *RACAL-PS*, although *RACAL-PS* achieves 2% higher specificity (on average) than *NNCA-PS*, it gives 6% less sensitivity (on average) than *NNCA-PS*. For soft classification as shown in Table 5.4, at 95% specificity, the proposed *NNCA-PS* achieves 90.8% and 86.7% sensitivity for normal and abnormal images respectively. When compared with other methods at the same specificity, the proposed *NNCA-PS* achieves 5.5% and 4.2% higher sensitivity than *RACAL-PS* and *KNN* respectively in case of normal images. Also in abnormal images at 95% specificity, *NNCA-PS* achieves 5.7% and 10.5% higher sensitivity than *RACAL-PS* and *KNN* respectively.

In order to investigate the difference between the proposed *NNCA-PS* and *RACAL-PS*, it is better to examine the clustering behaviour of both methods. *RACAL* achieves clustering using a radius parameter δ_0 which acts as the determinant of the cluster and controls the size of the clusters, which is the main constraint of *RACAL*, while *NNCA* achieves clustering without any control of cluster sizes as described in Section 5.2. This is demonstrated through experiments (see Section 5.4.1) as demonstrated in Table 5.3, where *RACAL-PS* achieves better specificity at the expense of the sensitivity compared with *NNCA-PS* where only 2% higher specificity gives 6% less sensitivity than *NNCA-PS*.

By comparing *NNCA-PS* with other classification methods for classifying breast tumors into either malignant or benign. The best classification accuracy is achieved by *NNCA-PS* (99.5%), with the lowest being 88.7% obtained by *PCA/MDC* as demonstrated in Table 5.5. Based on 100 experiments, although the classification accuracy of *GP/MDC* is comparable with *NNCA-PS*, it gives 0.6% less than the best performance of *NNCA-PS* with higher standard deviation in classification accuracy. Additionally, *GP/MDC* requires a nonlinear

feature generation, while the proposed *NNCA-PS* does not. This indicates the robustness of the proposed *NNCA-PS* compared with other sophisticated methods.

5.7 Summary

In this chapter, a novel clustering algorithm (*NNCA*) has been proposed, which achieves clustering without any control of cluster sizes. In addition, *NNCA* has augmented with a partial supervision (*PS*) strategy to act as a classifier. As shown, the proposed *NNCA-PS* has the ability to classify pixels of retinal images into those belonging to blood vessels and others not belonging to blood vessels, and it also has the ability to classify breast tumors into either benign or malignant. Experimental results show that the *NNCA-PS* offers better classification accuracies compared with other classifiers. Additionally, a parallel version of *NNCA* (*P-NNCA*) is proposed. As demonstrated, *P-NNCA* is scalable in terms of speedup and scaleup, which gives the ability to handle large datasets of high dimensions in a reasonable time. Moreover, *P-NNCA* is enhanced with a fast nearest neighbour procedure which helps in further reduction in the processing time.

Chapter 6

Investigations on Clustering Microarray Data

6.1 Introduction

Microarrays are a comparatively new technology to investigate the expression levels of thousands of genes simultaneously. Microarrays present new statistical problems because the data is highly dimensional with very little replication. Microarrays offer an exciting entry point for statisticians and computational scientists into the modern areas of computational biology and bioinformatics.

Gene expression data measured by microarrays are preprocessed using image analysis techniques to extract expression values from images and scaling algorithms to make comparable expression values. Expression data are typically analysed in matrix form with each row representing a gene and each column representing a sample. There are two types of clustering relevant to microarrays; the first is gene clustering (i.e., finding sets of genes with similar expression patterns), while the second is sample clustering (i.e., finding which samples are similar in terms of similarly expressed genes). In gene clustering, genes are treated as objects and samples are treated as features or attributes for clustering. Similarly, sample clustering treats samples as objects and genes as features or attributes. Cluster analysis for grouping functionally similar genes, gradually became popular after the application of the

average linkage hierarchical clustering algorithm for the expression of budding yeast and reaction of human fibroblasts to serum by Eisen *et al.* [96]. Herwig *et al.* developed a variant K-means algorithm to cluster a set of 2,029 human cDNA clones and adopted mutual information as the similarity measure [97]. Tomayo *et al.* [98] made use of SOM to cluster gene expression data. Since many genes usually display more than one function, Dembélé *et al.* [99] developed a fuzzy c-means clustering method for exposing these relations. Of course there exist many clustering algorithms already but the demand and structures of biological data are very different from those that have been studied using existing clustering algorithms. There are four important differences in the post-genomic, biological data. A standard requirement for existing clustering algorithms is to specify the number of clusters. In the present context, one does not know the number of clusters in these datasets; this lack of knowledge constrains the applicability of these algorithms. While in standard scenarios, all data belong to one or more clusters, only a small proportion of genes belong to clusters. Furthermore, the size of the datasets can be very large and yet statistics of specific examples can be rather low. All these point towards new techniques for clustering. Recently, novel clustering algorithms are developed to alleviate the requirement of the number of clusters. These algorithms include *SOON* (Self-Organising Oscillator Network) [100], and *RACAL* algorithm [95, 101] . In addition, there is no existing guideline for accepting one set of clustering results over another in extracting useful biological information from a particular dataset.

In this chapter, the use of a recently developed clustering algorithm, *SOON* that does not require the knowledge of the number of clusters, in analysing microarray datasets is investigated in Section 6.2. In addition, a novel integrated clustering performance measure (*ICPM*) is proposed to assess the reliability of results from a clustering algorithm used in analysing microarray data (see Section 6.3).

6.2 Self-Organising Oscillator Network (SOON)

6.2.1 Theory

6.2.1.1 Basic principles

The SOON algorithm, first proposed by Rhouma and Frigui [100], has its roots in a number of different biological processes that share the same physical characteristics. Fireflies flash at random when considered by themselves; however, when in large groups, fireflies exhibit the characteristic of firing together when in groups that are physically close to each other. Groups which are separated by distance will fire as disparate groups, each synchronised within itself. Heart pacemaker cells also share a similar behaviour, along with the menstruation cycles of groups of women in close proximity to each other. This behaviour, of self-organisation of components with an oscillatory nature, gives rise to the name of the algorithm - the Self Organising Oscillator Network (SOON).

6.2.1.2 Oscillator basics

The basic unit of clustering under the SOON algorithm is the Integrate and Fire (IF) oscillators which are simply described by some real valued state variable-representing for example a membrane potential- monotonically increasing up to a threshold. When this threshold is reached, the oscillator relaxes to a basal level by firing a pulse to the other oscillators, and a new periods begins. Mathematically, an oscillator can be defined using the following Eq. [100]:

$$x_i = f_i(\phi) = \frac{1}{b}\ln[1 + (e^b - 1)\phi] \tag{6.1}$$

where b is a constant value used to control the curve of the oscillator. Positive values of b will make the curve concave down, while negative values will make the oscillator concave up. $\phi \in [0, 1]$ is the phase angle of the oscillator, and determines the likelihood of the oscillator to fire, where 0 is just fired and 1 is firing. The output of the oscillator, x_i is bounded in the range $[0, 1]$, for all values of $f_i(\phi)$. This is achieved using a limiting function

$$B(x) = \begin{cases} x & \text{if} \quad 0 \leq x \leq 1 \\ 0 & \text{if} \quad x < 0 \\ 1 & \text{if} \quad x > 1 \end{cases}$$

Returning to the principle of the fireflies, it was noted that those which were physically close to each other would over time synchronise together to fire as one. This will require that the clustering algorithm will adjust the phase of individual oscillators such that the oscillators which are physically close will gradually take on first similar and then synchronised phase. Some form of adjustment of the phase values will be required in order to allow this process to occur.

At the end of each iteration of the training algorithm, the output value of an oscillator will change to a new value $x_j(t^+)$, using the formula

$$x_j(t^+) = B(x_j(t) + \epsilon_i(\phi_j)) \tag{6.2}$$

where $\epsilon_i(\phi_j)$, the coupling strength of an oscillator i at a given phase ϕ_j, is perhaps the most important concept of the whole algorithm. At this stage, adjustments are made to the state variables (and hence, ultimately the phase values) by applying an adjustment which considers the distance an oscillator is from the winning oscillator. Those oscillators physically near to the winning oscillator are made more likely to fire at the same time as the winning oscillator by adjusting the phase *towards* that of the winning oscillator, while those further away have the phase values adjusted so as to push them away *down* the phase curve, away from the winning oscillator. Depending upon the individual problem under consideration, the exact formulation of the coupling function used to calculate these adjustment values may vary; however, for the problem under consideration in this chapter, the following coupling function is proposed to calculate the coupling strength for oscillator i, ϵ_i, when the distance d_{ij} is measured against the winning oscillator ϕ_j.

$$\epsilon_i(\phi_j) = \begin{cases} C_E[1 - (\frac{d_{ij}}{\delta_0})^2], & \text{if} \quad d_{ij} \leq \delta_0 \\ -C_I[(\frac{d_{ij}-\delta_0}{\delta_1-\delta_0})^2], & \text{if} \quad \delta_0 < d_{ij} \leq \delta_1 \\ -C_I & \text{otherwise} \end{cases} \tag{6.3}$$

Having decided on a limit distance δ_0, δ_1 is set to be five times δ_0. The coupling function promotes all oscillators which lie within the distance δ_0, increasing the phase value by the constant of excitation, C_E, multiplied by a factor depending on the ratio of the distance between the winning oscillator and the oscillator under consideration, and δ_0. This in turn makes the group of oscillators more likely to synchronise with the winning oscillator in future iterations. The phase of all those with distance lying in the interval $\delta_0 < d_{ij} \leq \delta_1$ are inhibited by C_I, the coefficient of inhibition multiplied by a ratio that takes account of how close the oscillator under consideration is to the winning oscillator. All values of $d_{ij} > \delta_1$ are hard limited to $-C_I$. C_E is typically relatively small, of the order 0.1-0.2. C_I is normally set to the value C_E/N (N is the number of data points, objects, in the training set), as any given data point is likely to be inhibited more often than it is likely to be excited.

6.2.1.3 SOON-1 algorithm

The SOON-1 algorithm is the most basic form of the SOON algorithms. In this form, all training points start off as being the initial set of oscillators. All oscillators use a spheroid enclosure of the same radius, which is set as a parameter on commencement of the algorithm. The algorithm determines which of the oscillators is the next to fire by examining individual phases of each oscillator and selecting the one with phase ϕ_i closest to 1, and increases the phase of all oscillators by a set amount $(1 - \phi_i)$. Having adjusted all phases as necessary, the state x_i of each oscillator is calculated using Eq. 6.1. From the state variables, the distances to winning oscillator are updated and then the coupling strengths (ϵ_i) are calculated using the coupling function in Eq. 6.3, which allows the state variables to be adjusted based on the distance of each oscillator to the winning oscillator. The state variables are then adjusted using Eq. 6.2, giving the revised output values. Having calcu-

lated the revised state variables (which take account of the proximity coupling effect ϵ_i), the new phase values can be calculated using the inverse of the oscillator function:

$$\phi_j = f^{-1}(x_j) \tag{6.4}$$

At the end of this cycle of the algorithm, points which were physically close to the oscillator which fired will move closer together, gradually tending to synchronise, whilst those which were further away will move away from the winning oscillator. As other different oscillators fire in different iterations of the algorithm, then these in turn will start to synchronise with their closest neighbours. Alternatively, certain oscillators will be too far away from any others to form a cluster, and will essentially remain as individual clusters.

6.2.1.4 SOON-2 algorithm

One potential problem with the SOON-1 algorithm is that it requires the calculation of a large distance matrix which records the distance of every point in the training set from every other point, where SOON-1 algorithm uses all training points as initial oscillators. By reducing this number to a lower value, and distributing the reduced number of points over the data space, a series of *prototypes* can be created. The selection of the points may be either existing points in the training set, or alternately the points may be selected to highlight specific areas of interest in data-space, increasing the likelihood of clusters being created that shares the profiles of interest. This is of particular interest in certain areas, such as that of microarray analysis, where certain gene expression profiles may be of interest due to biological or physiological processes that are thought to be of significance in a particular operation. Biologists can provide a certain profile that they believe would approximate the behaviour of interest, and then see whether any genes share a similar profile.

Cluster shape

The simplest cluster is a spherical cluster using a Euclidean distance measure; this is ideal for many scenarios - particularly in the case of QAM-4 communication data, where the clusters are often spherical. Many other data types do not naturally form spherical clusters.

Ellipsoidal clusters better map the distribution of data in these cases where the Mahalanobis distance measure [12] can be more appropriate. Given a point $\mathbf{x}$, and series of vectors which form a cluster Cl with cluster mean $\bar{\mu}_{cl}$, the covariance matrix of the cluster Cl gives a measure of the variance in each dimension of the data under consideration. This in turn allows the ellipsoid to take on different "widths" in each dimension. In the case of an equi-variance cluster, the Mahalanobis distance will be equal to a Euclidean distance.

$$d^2(\mathbf{x}, Cl) = (\mathbf{x} - \bar{\mu}_{cl})\mathbf{C}_{cl}^{-1}(\mathbf{x} - \bar{\mu}_{cl}) \tag{6.5}$$

Having calculated the Mahalanobis distance, this is then normalised using a χ^2 distribution with p degrees of freedom, where p is the number of dimensions of the input dimension. This then gives a normalised Mahalanobis distance. A normalised Mahalanobis distance in conjunction with the coupling function given in Eq. 6.3 is proposed.

$$d^2(\mathbf{x}, Cl) = \frac{d^2(\mathbf{x}, Cl)}{5\chi^2(p, \alpha)} \tag{6.6}$$

where α is the percentile of inclusion of the χ^2 distribution. The value of α is varied to fit the size of the Mahalanobis cluster relative to the data.

6.2.2 Cluster validation

When the number of clusters present within the dataset is known a priori (for example, microarray liver cancer dataset contains two clusters, one representing benign samples and the other representing malignant samples), it is useful to verify the validity of the clustering results (i.e., how well the clustering algorithm discovered the clusters of the dataset). Some attempts have been made to estimate the number of clusters by evaluating the clustering results based on a unique validation index [102, 103]. Some other attempts have been made to combine different validation techniques to estimate the number of clusters [104, 105, 106]. Therefore a combination of different validation methods can be successfully used for the estimation the number of clusters. In this study, *DB* [56], *CH* [58], *I* [59], and *XB* [64] cluster validity indices are used to assess the clustering results obtained by SOON.

6.2.3 Datasets

In addition to the communication dataset which is used as a validation dataset, three microarray datasets have been used to examine the applicability and reliability of the SOON algorithms, along with some standard algorithms, in analysing microarray data.

Microarray yeast data

Microarray yeast data is taken from the Stanford Yeast Cell-Cycle Project [107]. The initial data consists of 17 observations taken at 10 minute intervals on approximately 6,400 different genes during the cell cycle process. A variation filter was used to eliminate genes that did not change significantly across samples. After normalisation and prefiltering in order to remove minimally variant genes, the number of genes available for consideration drops to just over 1,000 (1,002 genes). Therefore, the input data consists of 1,002 genes each of 17 observations. The number of clusters present within this dataset is not clearly defined, as the data is not clearly separable; however there are a number of groups present within the data that represent different biological processes within the cell cycle of the yeast organism. There have been a number of papers [98, 108, 109] which describe these clusters, along with their biological meanings.

Microarray lymphoma data

Microarray lymphoma data [110, 111] has three most prevalent adult lymphoid malignancies: Diffuse Large B-Cell Lymphoma (DLBCL), Follicular Lymphoma (FL), and Chronic Lymphocyte Leukaemia (CLL). To provide a framework for interpretation of the gene expression in these patient samples, gene expressions are profiled in purified normal lymphocyte subpopulations under a range of activation conditions, in normal human tonsil and lymph node, and in leukaemia cell lines [112]. Fluorescent cDNA probes, labelled with Cy5 dye, were prepared from each experimental messenger RNA sample. A reference cDNA probe, labelled with Cy3 dye, was prepared from a pool of mRNAs isolated from nine different lymphoma cell lines. Each Cy5-labelled experimental cDNA probe was combined with Cy3-labelled reference probe and the mixture was hybridised to the

microarray. The fluorescence ratio was quantified for each gene and reflected the relative abundance of the gene in each experimental mRNA sample compared with the reference mRNA. The use of a common reference probe allows the treatment these fluorescent ratios as measurements of relative expression level of each gene across all experimental samples. Approximately 1.8-million measurements of gene expression were made in 96 normal and malignant lymphocyte samples using 128 Lymphochip microarrays, and the total number of genes is 4,026 genes.

Microarray liver cancer data

Microarray liver cancer data [113, 114] has two classes: the first represents Hepatocellular carcinoma (HCC) samples, and the other represents non-tumor liver tissues. Primary data were carried out using GenePix Pro 3.0 (Axon Instruments). Areas of the array with obvious blemishes were manually flagged and excluded from subsequent analysis. The raw data were deposited into Stanford Microarray Database [115] at [114]. All non flagged array elements for which fluorescent intensity in each channel was greater than 1.5 times the local background were considered well measured. Genes for which less than 75% of measurements across all the samples in this study met this standard were excluded from all analysis. Therefore, one can get 3,180 genes (represented by 3,964 cDNAs) with the greatest variations in expression in 156 liver samples (74 non-tumor liver and 82 HCC).

6.2.4 Experiments and Results

At all stages, the coupling function was kept constant, as given in Eq. 6.3. The constant of excitation was set to 0.1, with $C_I = C_E/N = 0.1/N$ (N is the number data points available for clustering), and $b = 3$.

6.2.4.1 Communication data

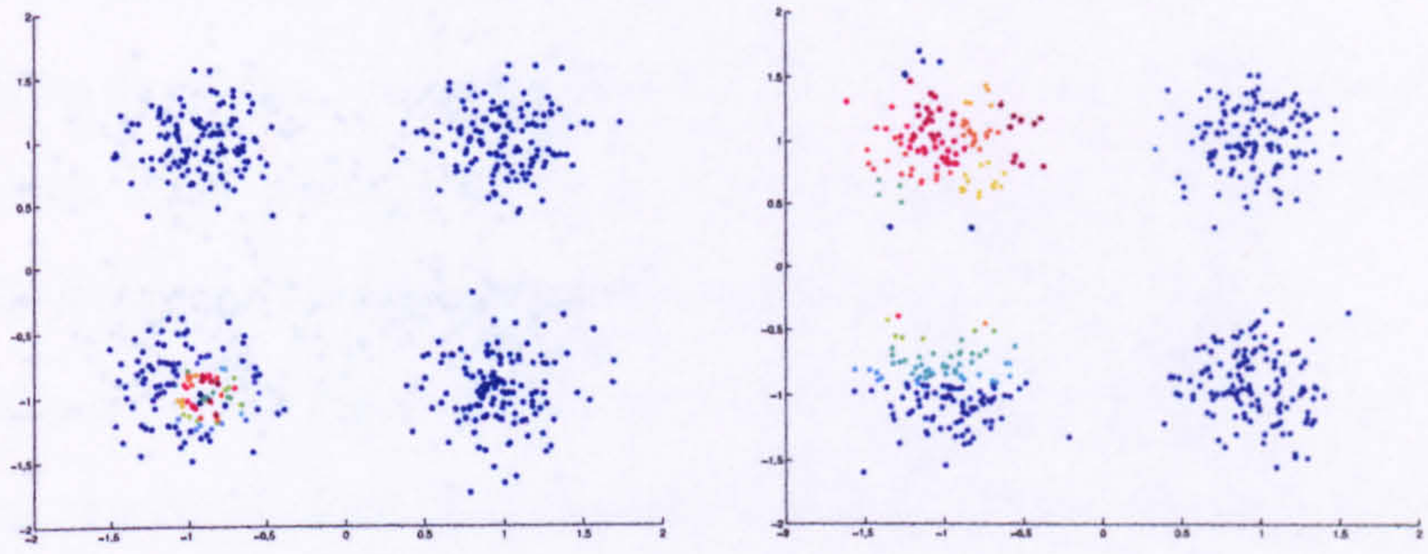
This dataset consists of 512 points of data, which is a typical training size for data in this application. Starting with an initial 512 cluster centres, then SOON-1 was allowed to train until the number of clusters stabilised, when the training was stopped. The number

of clusters and cluster centres were then compared with the constellation diagram for the correct signal. This was carried out for two different signal-to-noise ratio (SNR) values: 15 and 10 dB. For each SNR value, five different values of δ_0: 0.01, 0.06, 0.11, 0.16 and 0.21. This gave a total of 10 different experiments. Each SOON-1 was allowed to train for 350 training epochs, and then the results were examined.

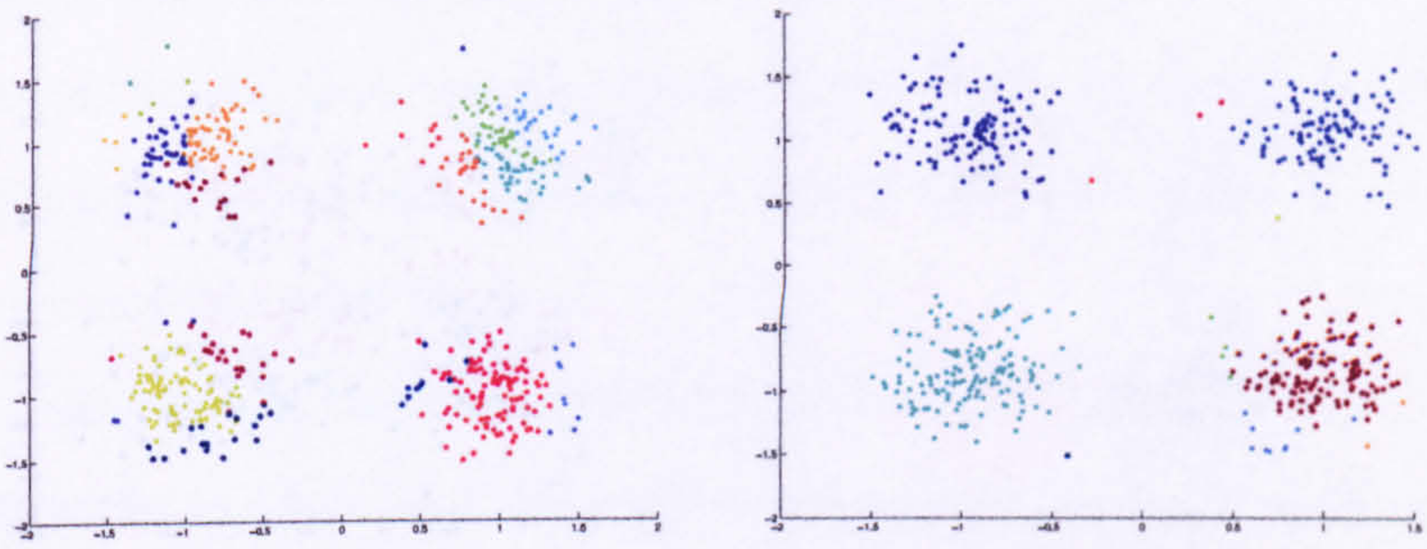
Figure 6.1 shows the clustering performance on a QAM-4 at SNR = 15 dB for five different values of δ_0. Each colour on the plot shows a different cluster. At the smallest value (0.01 - (a)), the radii of the clusters are insufficient to capture any significant membership of points, and as a result, the performance is very poor, with a lot of small single member clusters being created in one of the large clusters, while the other three large clusters are too far away from the centres to be enclosed in a cluster. As a result, most of the data is not assigned to a cluster. If the algorithm had been allowed to train longer, then the whole dataset would have become individual clusters, where the smallest value of δ_0 inhibits any oscillator to synchronise with the winning oscillator as described in Eq. 6.3.

As the value of δ_0 increases, the spread around each cluster increases, increasing the membership of each cluster, and encompassing more of the data. Figures 6.1(b) through to 6.1(e) show the clustering performance as the value of δ_0 is increased. At $\delta_0 = 0.01$ and 0.06, several points are not clustered. At $\delta_o = 0.11$, 0.16, and 0.21, all the data points are clustered to 23, 6, and 4 clusters respectively.

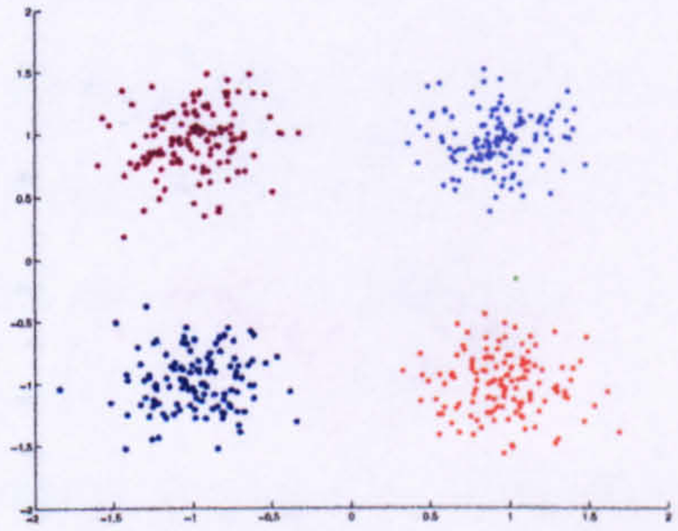
For communication data with SNR = 10 dB, Fig. 6.2 shows the results of the clustering performance. Once again, this shows that as the value of δ_0 is increased, the clustering becomes consistent and relatively well distributed through the data space. As δ_0 is increased further, all the clusters would gradually merge to form one large cluster. Of importance, however, is the fact that the clustering behaviour remains controlled over time; by exercising choice of th δ_0 parameter, it is possible to control the size of the clusters that are generated, even on data which is intrinsically non-separable in nature. The desirable behaviour of the SOON algorithm has been verified with a series of such validation sets with varying values of SNR. This bodes well for the biological data, and shows that the behaviour of the algorithm is consistent with the desired behaviour for non-separable data.



(a) $\delta_0 = 0.01$ (b) $\delta_0 = 0.06$



(c) $\delta_0 = 0.11$ (d) $\delta_0 = 0.16$



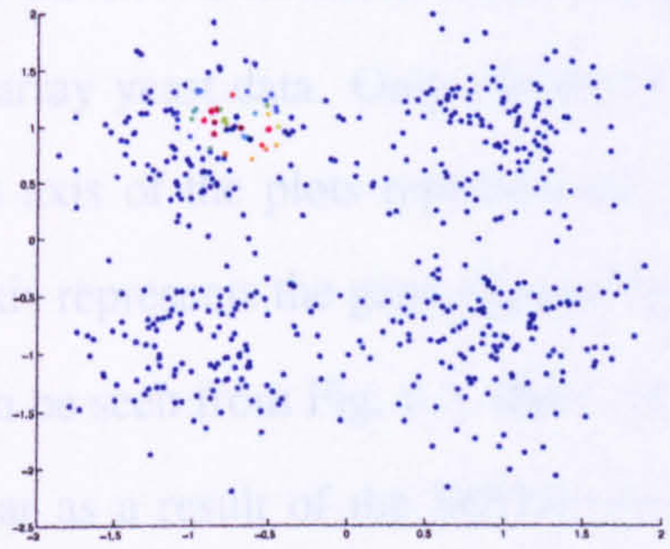
(e) $\delta_0 = 0.21$

Figure 6.1: Clustering performance for QAM-4 at SNR = 15 dB.

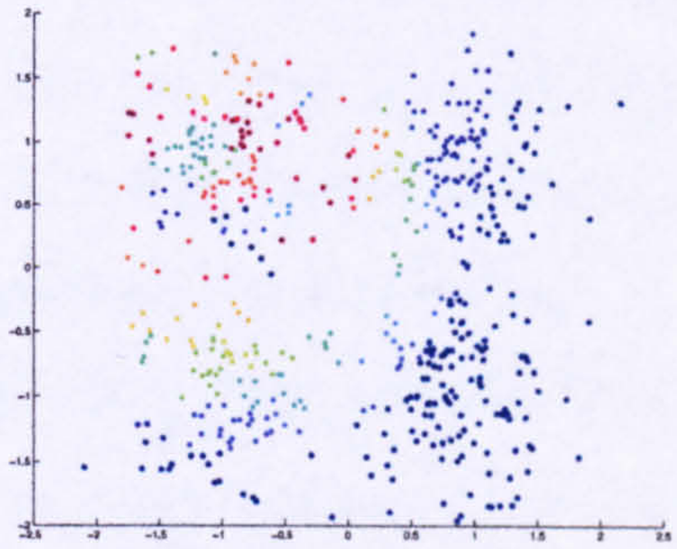
6.2.4.2 Microarray data

Microarray yeast data

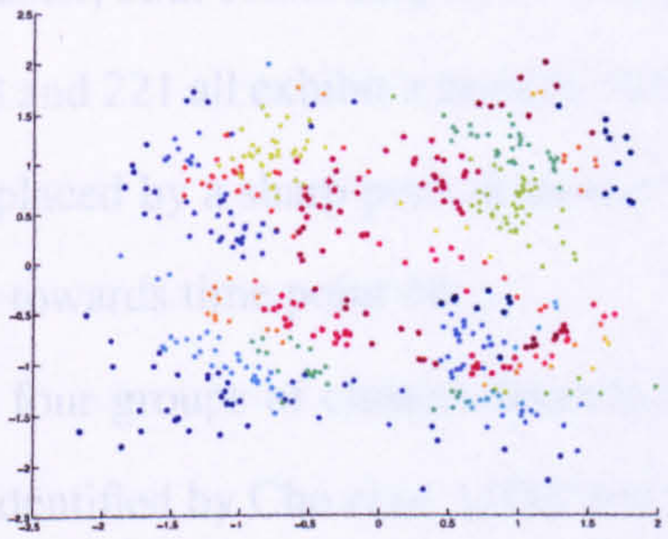
For microarray yeast data, SOON-2 was initially tested using Euclidean distance measure, giving spheroid clusters. Using randomly selected 501 prototypes, one half of the total number of data points available for clustering, the algorithm was allowed to stabilise, whereupon the clusters were examined. This process was then repeated using the Maha-



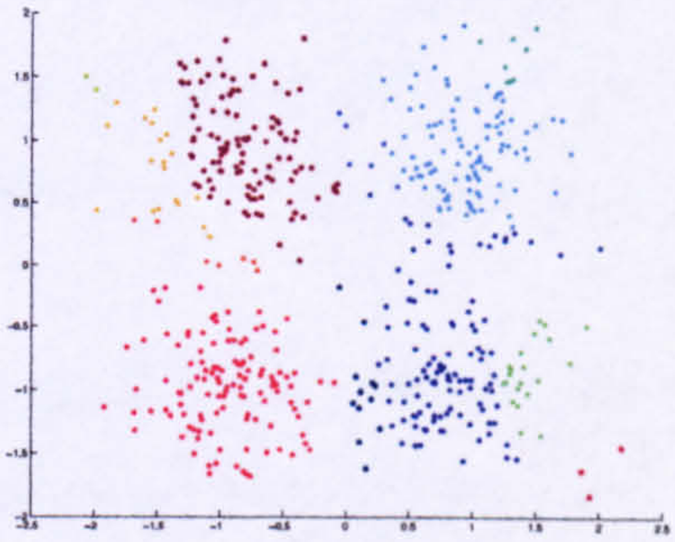
(a) $\delta_0 = 0.01$



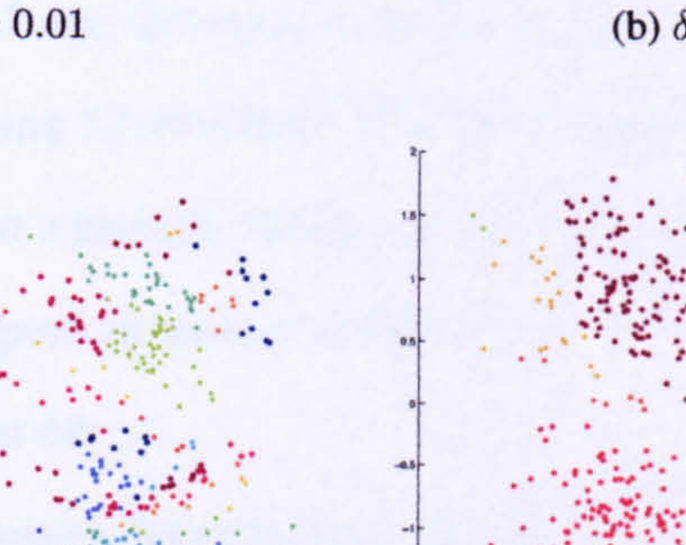
(b) $\delta_0 = 0.06$



(c) $\delta_0 = 0.11$



(d) $\delta_0 = 0.16$



(e) $\delta_0 = 0.21$

Figure 6.2: Clustering performance for QAM-4 at SNR = 10 dB.

lanobis distance measure. Any cluster with fewer than six members was disregarded.

For the Euclidean distance experiments, the value of δ_0 was allowed to vary between 0.01 and 0.29 in increments of 0.02. The value of both α and δ_0 were varied for the Mahalanobis distance measure experiments, with α ranging between 0.01 and 1.01 in increments of 0.1, and δ_0 varying between 0.01 and 0.21 in increments of 0.05. This allowed an examination of the effect of the two different parameters upon the clustering performance of the algorithm, particularly with regard to non-separable data.

Figure 6.3 shows results of a clustering run using the Euclidean distance measure and the microarray yeast data. Only clusters with more than six members are displayed. The horizontal axis of the plots represent the time course in ten minute intervals, while the vertical axis represents the gene expression magnitude after normalisation.

As can be seen from Fig. 6.3, there are a number of different clusters that make themselves clear as a result of the SOON analysis. Clusters 9 and 177 both show the same characteristics, containing 67 and 49 members respectively. Clusters 36 and 132 also form a larger cluster, both containing 12 members. Clusters 17 contains 41 members, while clusters 6, 118 and 221 all exhibit a broadly similar general trend where a low initial expression level is replaced by a sharp peak at around timepoint 10, followed by a decrease, and then a increase towards time point 16.

These four groups of clusters broadly correspond to phases G1, S, G2 and M respectively, as identified by Cho *et al.* [109] and Spellman *et al.* [108]. Additionally, the clusters also match fairly closely in shape to those identified by Tamayo *et al.* [98].

Of interest are clusters 100 and 112 on the figure which describe a small cluster of rather distinctive behaviour; the gene remains predominantly stable around level 0 for most of the experiment, however at timepoint 10 there is a sudden trough to approximately -3 in magnitude. This occurs at the same time as several other groups of genes are peaking in the opposite direction.

It is interesting to note that in Tamayo *et al.* [98], none of the clusters shown as the output of the SOM show a similar pattern. This may in part be due to the fact that the variation filter used in these experiments gave different results to that of Tamayo, selecting 1,002 genes rather than the 823 used in their experiments. This may explain why this group of clusters (100 and 112) appears in one analysis and not in another, however carrying out tests using a MATLAB based SOM toolbox on the same dataset (Fig. 6.4) with initial geometry of nodes in 6 x 5 grid as in Tamayo *et al.* [98] also failed to highlight this relatively small cluster as a point of interest. The effect of varying the geometry of the SOM does not produce fundamentally new patterns as demonstrated in Tamayo *et al.* [98]. With only a few nodes, no distinct patterns are obtained and there is a large within-cluster

scatter. As nodes are added, distinctive and tight clusters emerge. Beyond this point, the addition of further nodes tends to produce no fundamentally new patterns. As shown in Fig. 6.4, only cluster 6 is the closest match to the two clusters, however the depth of the dip in the profile is much wider, and also less pronounced than for the SOON.

When using the Mahalanobis distance [12] measure instead of the Euclidean distance, having two independent control parameters gives some degree of flexibility as to the configuration of the algorithm. Experimentation with the two parameters, δ_0 and α, tended to suggest that small values of δ_0 (< 0.05) will cause a very high number of very small and tight clusters to be created, very often with only one or two members. Increasing the value of δ_0 increased the size of the membership of each cluster, while still tending to keep the variance within cluster membership relatively tight. Increasing the value of δ_0 to large values meant that the clusters became so large that all the clusters held no clearly discernible significant details.

The effect that the variation of the α parameter values were having upon the performance of the algorithm was less clear; selecting a value of α effectively moves the cut-off point used to determine membership of the cluster by scaling the distribution of the data.

Adjustment of each parameter affects the performance of the clustering algorithm in different ways. Adjustment of α controls the "tightness" of the cluster, while δ controls the size of the clusters through the radius parameter of the cluster. This is demonstrated through the second set of experiments using the Mahalanobis distance measure. Figures 6.5 through 6.8 show the effect of increasing the α value while keeping the δ_0 value fixed.

In Fig. 6.5, where $\alpha = 0.01$, or only one percent of the confidence interval, clusters 6, 173 and 205 show three basically similar clusters which correspond to the G1 phase of the cell cycle, containing 53, 51 and 19 members respectively. In Fig. 6.6, the α value has been increased to 0.11, and the cluster numbers with this general profile are 30 and 74, with 28 and 88 members respectively. Increasing the α value again to 0.21, Fig. 6.7 shows clusters 144 and 224, with 103 and 14 members, while Fig. 6.8 shows the transition of membership to one single cluster 159 with 111 members. These results are summarised in Table 6.1. It is observed from Table 6.1 that numbers of members (genes) of G1 phase with similar

Table 6.1: Cluster size and distribution for varying sizes of α, when $\delta_0 = 0.21$.

	Cluster no. (membership)				Number of members			
α	G1	S	G2	M	G1	S	G2	M
0.01	6 (53) 173 (51) 205 (19)	30 (9) 149 (22) 191 (17)	115 (12) 182 (10) 238 (20)	20 (14) 39 (15) 113 (12) 118 (8) 248 (20)	123	48	42	69
0.11	30 (28) 74 (88)	144 (19) 229 (12) 245 (14)	51 (19) 171 (9) 179 (20)	45 (20) 123 (41)	116	45	48	61
0.21	144 (103) 224 (14)	52 (13) 139 (21) 249 (9)	55 (32) 120 (13)	65 (13) 167 (28) 193 (6) 238 (7)	117	43	45	54
0.31	159 (111)	181 (26) 198 (12)	61 (6) 82 (16) 165 (22)	11 (6) 133 (55)	111	38	44	61

Table 6.2: Detailed memberships of similar clusters for G1 phase at different α values.

$M_{\alpha=0.01} \cup M_{\alpha=0.11} \cup M_{\alpha=0.21} \cup M_{\alpha=0.31}$	138
$M_{\alpha=0.01} \cap M_{\alpha=0.11} \cap M_{\alpha=0.21} \cap M_{\alpha=0.31}$	100

behaviour at α values of 0.01, 0.11, 0.21 and 0.31 are respectively 123, 116, 117 and 111. Let these be represented by $M_{\alpha=0.01}$, $M_{\alpha=0.11}$, $M_{\alpha=0.21}$, and $M_{\alpha=0.31}$. It is clear that as the values of α changes within this range, the number of members does not change much. This is a desirable quality of a cluster algorithm.

Furthermore one can investigate the detailed memberships of these similar clusters at different values of α. To this end, the union of memberships at these values of α is calculated, and that gives a total of only 138 genes that are involved in all these clusters found at different α values. Also, by calculating the intersection of these members to discover that one hundred members out of the 138 members are common to all clusters at different α values. These are presented in Table 6.2. It is clear from Tables 6.1 and 6.2 for the G1 phase of the cycle the relevant membership is very robust with respect to the choice of the α parameter values (varying by a factor of 30, from 0.01 to 0.31). Such robustness is an important and desirable characteristic of an algorithm. Another example of robustness is the interesting observation that the pattern of cluster 55 in Fig. 6.5 ($\alpha = 0.01$) is also present in cluster 8 ($\alpha = 0.11$), cluster 42 ($\alpha = 0.21$), cluster 75 ($\alpha = 0.21$) and cluster 74

Table 6.3: Evaluation of different combinations of α and δ parameters using different validity indices.

	Validity index	$\alpha = 0.01$	$\alpha = 0.11$	$\alpha = 0.21$	$\alpha = 0.31$
$\delta_0 = 0.01$	I [59]	33.97	**40.20**	6.69	34.58
	CH [58]	647.48	**881.36**	279.48	684.47
	XB [64]	0.055	**0.036**	0.075	0.038
	DB [56]	0.303	**0.24**	0.479	0.25
$\delta_0 = 0.06$	I	4.60	0.87	3.17	1.05
	CH	181.91	98.64	120.25	112.44
	XB	0.106	0.504	0.289	0.459
	DB	0.373	0.484	0.417	0.456
$\delta_0 = 0.11$	I	0.274	0.468	0.330	0.748
	CH	46.24	59.65	53.51	71.16
	XB	2.39	0.53	0.79	0.24
	DB	0.74	0.56	0.68	0.50
$\delta_o = 0.16$	I	0.06	0.15	0.07	0.10
	CH	25.10	32.42	26.44	27.08
	XB	10.79	3.51	5.23	4.34
	DB	1.13	0.85	1.06	1.02
$\delta_0 = 0.21$	I	0.014	0.017	0.013	0.020
	CH	17.52	19.99	18.35	21.78
	XB	14.01	34.88	16.52	4.80
	DB	2.16	2.03	1.87	1.50

($\alpha = 0.31$). These generalised cluster shapes are held across both the Euclidean distance and Mahalanobis distance measures; again, the algorithm is fairly robust with respect to the choice of the distance function. Using the Mahalanobis distance measure offers more control than the Euclidean measure; however this is at the cost of increased complexity in the calculations. It is also important to note that the SOON is capable of finding all the same clusters as the SOM, however with the added bonus of highlighting smaller clusters that are swamped by the higher density of other clusters in close proximity when clustering using the SOM.

Conversely, results for keeping the α value constant and changing the δ_0 value have been obtained but are not presented here. As the value of δ_0 is increased ($\delta_0 = 0.01$, 0.06, 0.11, 0.16, 0.21), the number of clusters with high membership (> 6 members for the purposes of this work) increases. Table 6.3 shows the evaluation of SOON clustering results at different combinations of α and δ_0 using different validity indices. As shown, the best combination of α and δ_0 is achieved at $\alpha = 0.11$ and $\delta_0 = 0.01$ where I and CH validation indices are maximised and XB and DB validation indices are minimised. In this

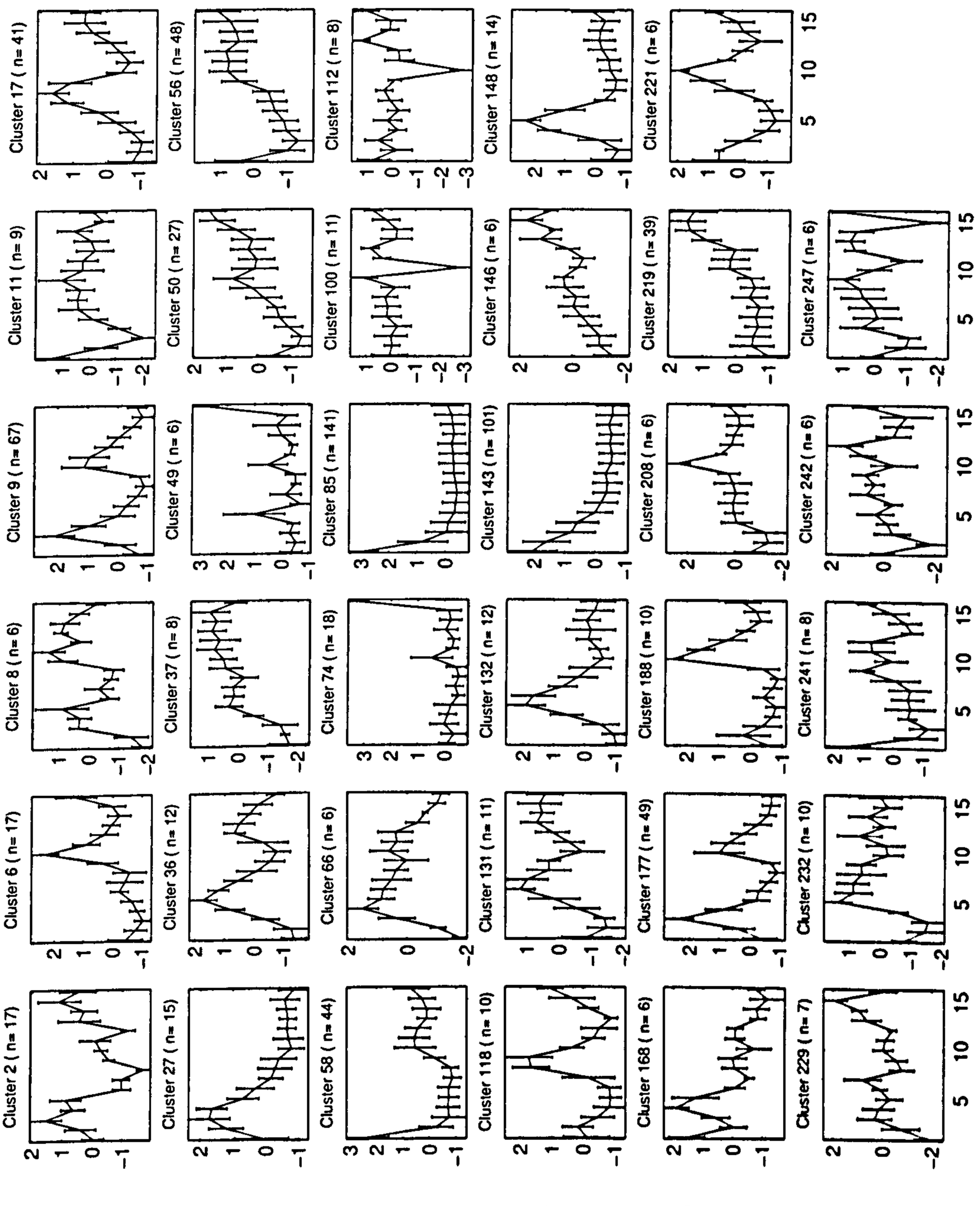


Figure 6.3: Clustering results using the Euclidean distance with $\delta_0 = 0.2$ on the microarray yeast data.

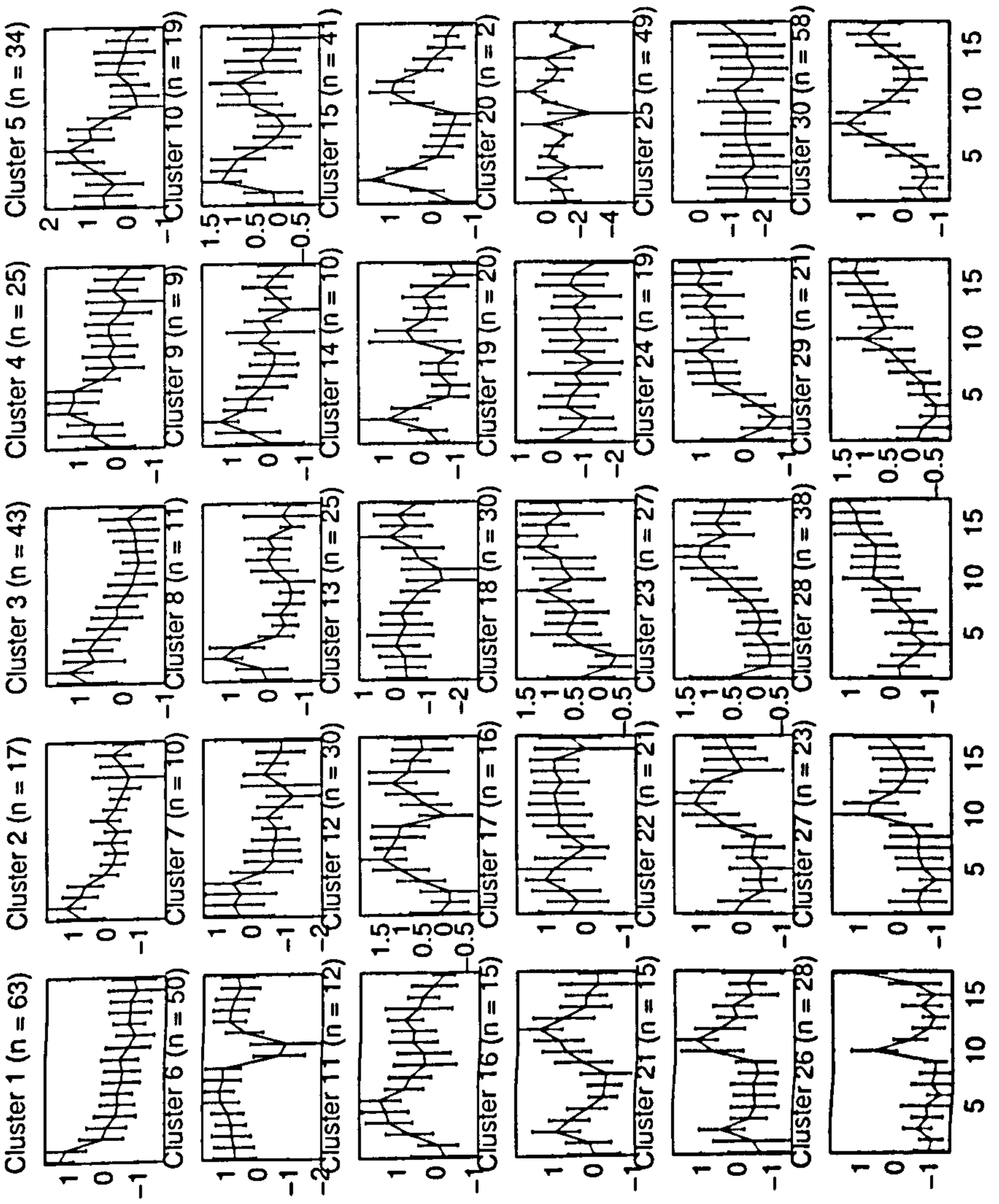


Figure 6.4: Clustering results using SOM.

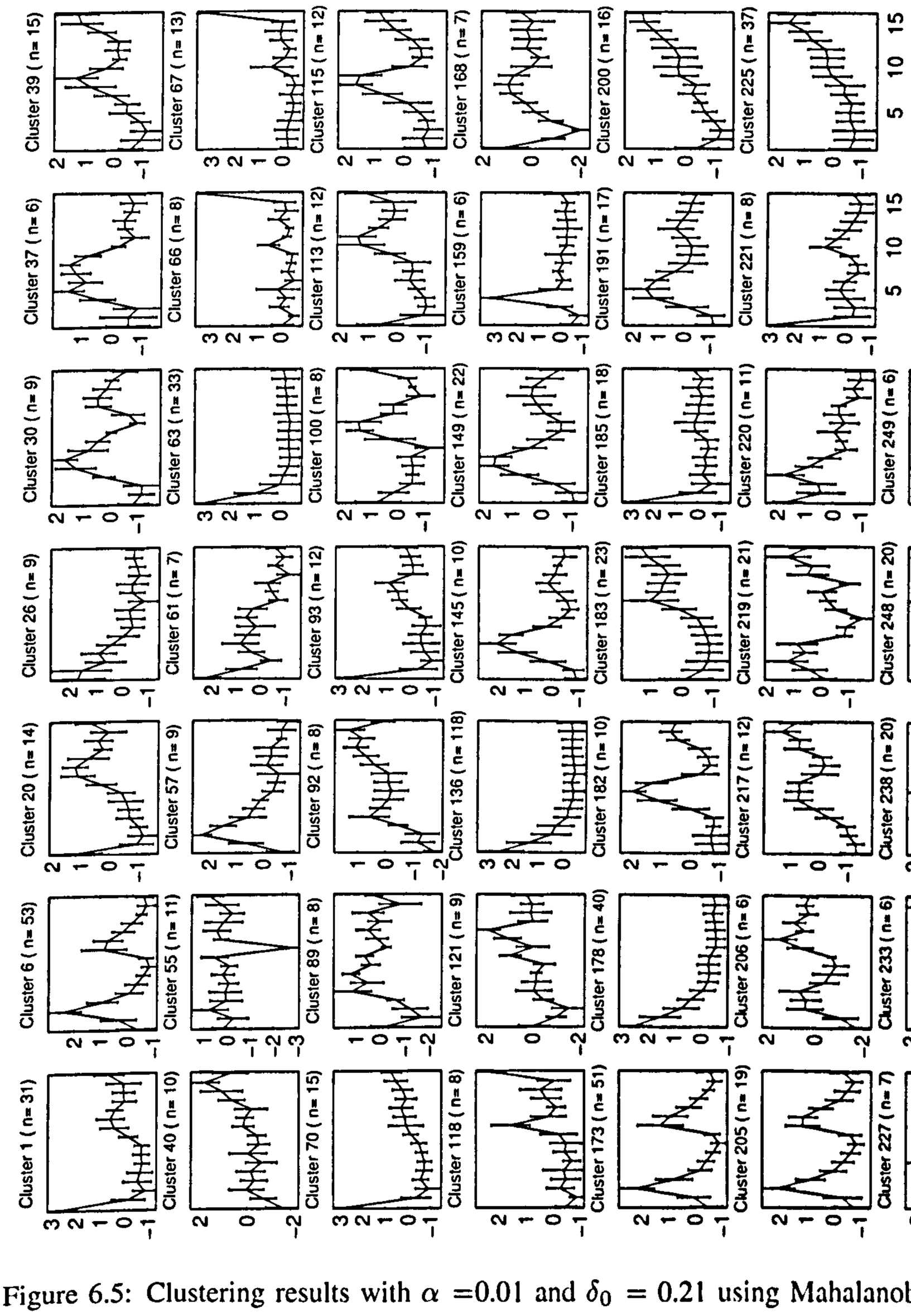


Figure 6.5: Clustering results with α =0.01 and δ_0 = 0.21 using Mahalanobis distance measure.

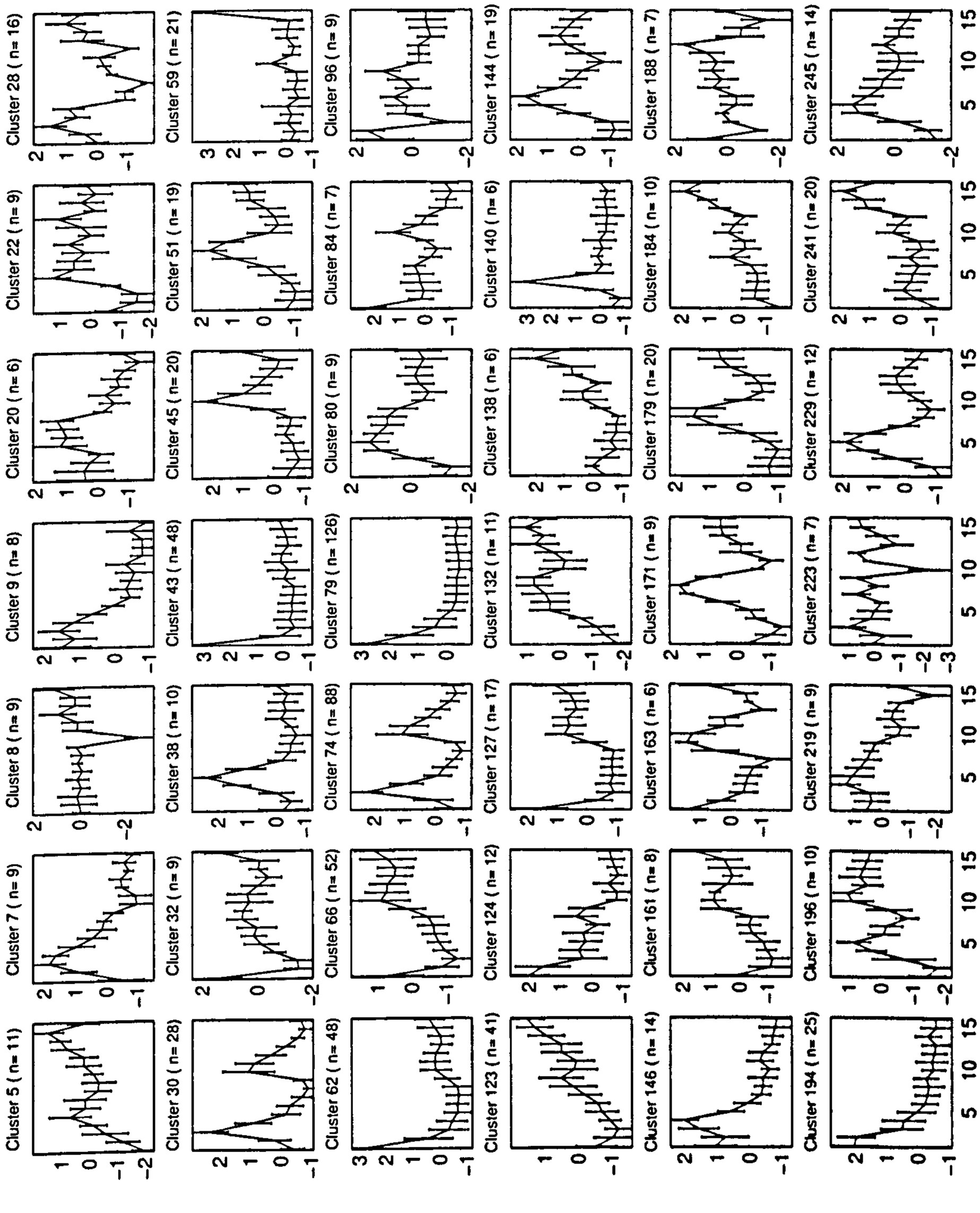


Figure 6.6: Clustering results with $\alpha = 0.11$ and $\delta_0 = 0.21$ using Mahalanobis distance measure.

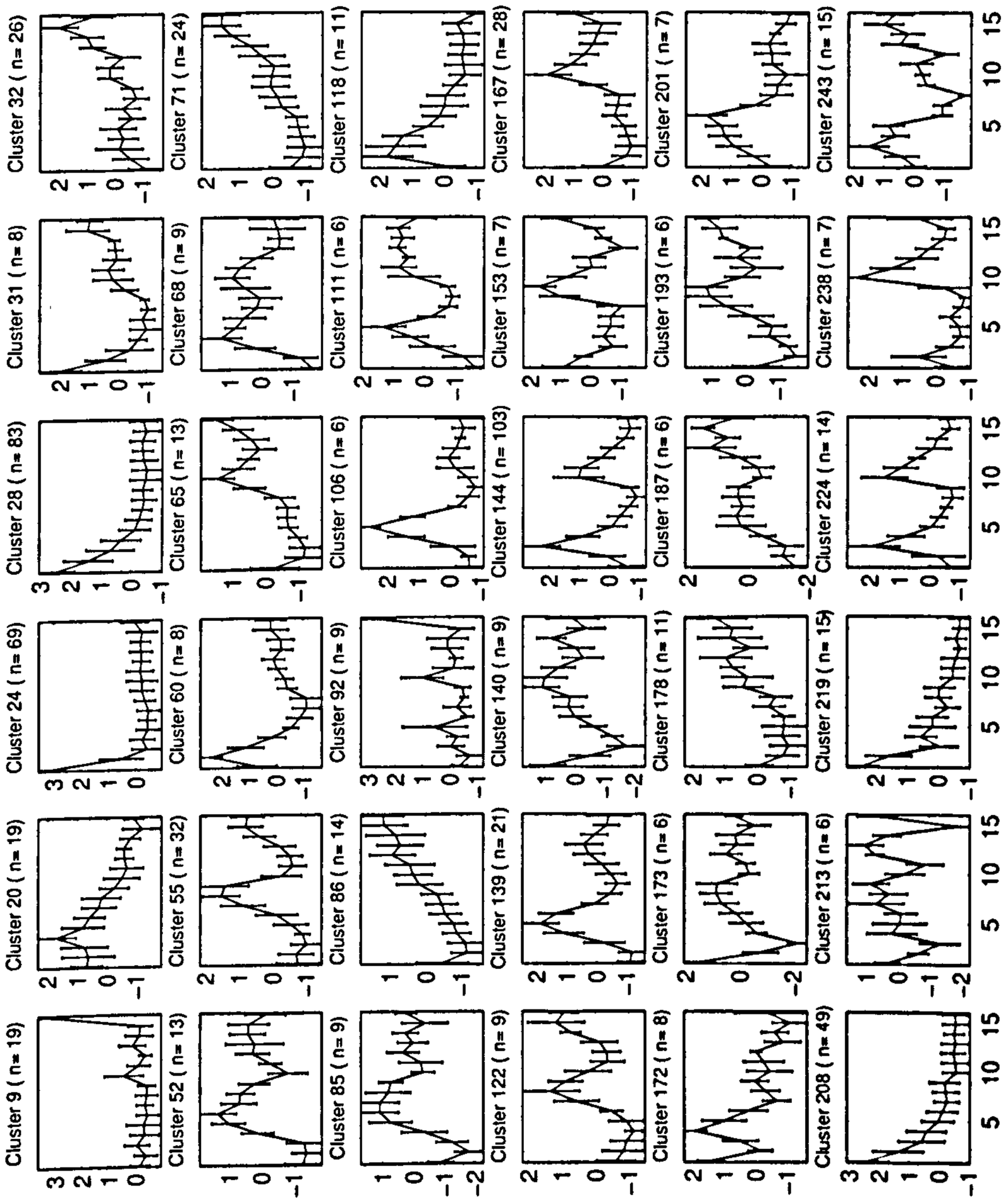


Figure 6.7: Clustering results with $\alpha = 0.21$ and $\delta_0 = 0.21$ using Mahalanobis distance measure.

Figure 6.8: Clustering results with $\alpha = 0.31$ and $\delta_0 = 0.21$ using Mahalanobis distance measure.

way, through the choice of the values of α and δ_0, it is possible to modify the output of the clustering algorithm for achieving proper clustering result. Finally, SOON offers a flexible degree of control in the granularity of the clusters, which can be exercised over cluster formation.

Microarray cancer data

For microarray cancer data, SOON-2 is applied along with Euclidean distance using random prototypes set to one half the total number of data points available for clustering (i.e., 48 and 78 prototypes for lymphoma and liver cancer data respectively). For each data, a series of experiments were carried out using SOON-2 at different δ_0 values (i.e., one can get different clustering partitions). Then, *DB* [56], *CH* [58], *I* [59], and *XB* [64] and cluster validity indices are used to assess the clustering results which are obtained by SOON-2 algorithm, and checking its ability to bring out the inherent structures in the dataset and discover the clusters of the dataset.

A series of experiments at different δ_0 values were carried out to investigate the clustering behaviour of SOON algorithm on microarray cancer datasets, and its ability to bring out the inherent structures in the datasets and discover the clusters in the datasets.

Figure 6.9 contains two subplots: the topmost subplot shows the validation indices values versus the radius (δ_0), while the bottom subplot shows the number of clusters K corresponding to the same δ_0 values. As shown, the change of validity index appears as a *"knee"* in the plot and it is an indication of the number of clusters underlying the data (i.e., more compact clusters and wider cluster separations between clusters). As shown in Fig. 6.9, the number of clusters corresponding to the "knee" points for which *I* and *CH* validation indices are maximised and *XB* and *DB* validation indices are minimised, is $K = 3$. The value of $K = 3$ represents the best fit number of clusters which actually coincides with the true number of clusters. The result of sample clustering suggests that genes in the same cluster have similar functions, or they share the same transcriptional regulation mechanism. From biological and clinical points of view, identifying similar genes can help medical researchers to investigate the mechanisms for cancer development

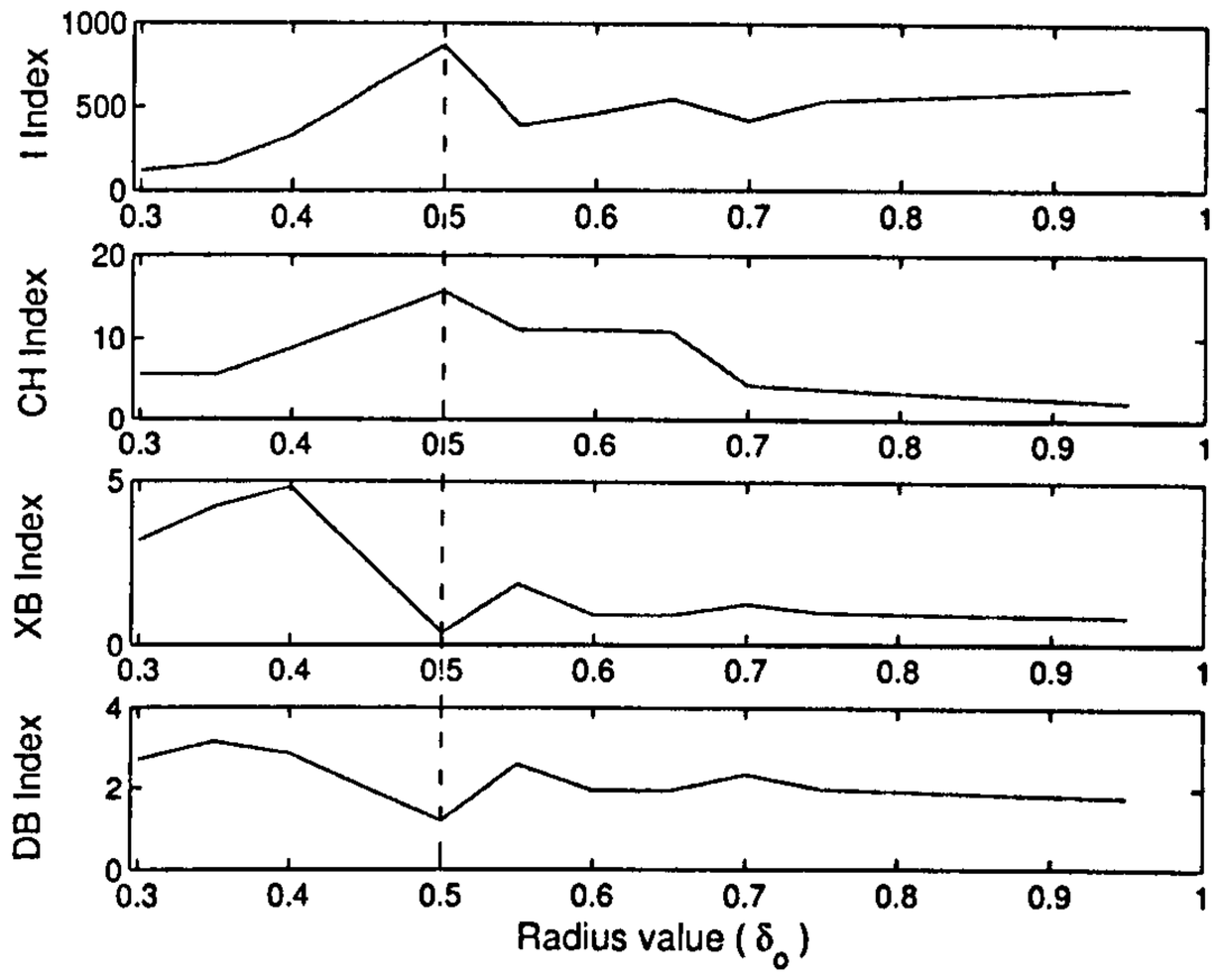
Table 6.4: Comparisons with other clustering algorithms for lymphoma cancer data.

Number of clusters	Validity index	Hierarchical	SOM	K-means	SOON
$K = 2$	I [59]	**700.00**	**610.01**	**612.25**	604.06
	CH [58]	**11.00**	**4.83**	**5.09**	2.22
	XB [64]	**1.04**	**1.59**	**1.45**	0.85
	DB [56]	**1.12**	**2.03**	**1.62**	1.79
$K = 3$	I	613.05	400.63	436.93	**862.04**
	CH	7.84	4.19	4.61	**15.72**
	XB	1.37	2.39	2.08	**0.37**
	DB	1.24	2.25	2.09	**1.18**
$K = 5$	I	298.70	178.62	201.51	328.66
	CH	4.77	2.93	3.35	8.82
	XB	4.97	6.82	5.64	4.82
	DB	2.93	3.78	3.18	2.87
$K = 6$	I	126.22	80.87	85.07	155.97
	CH	4.52	1.94	2.12	5.52
	XB	4.82	6.00	5.72	4.22
	DB	3.39	3.91	3.62	3.16
$K = 9$	I	110.71	56.52	64.87	117.62
	CH	3.56	1.61	1.87	5.57
	XB	3.30	4.48	4.00	3.20
	DB	2.84	4.29	3.71	2.72

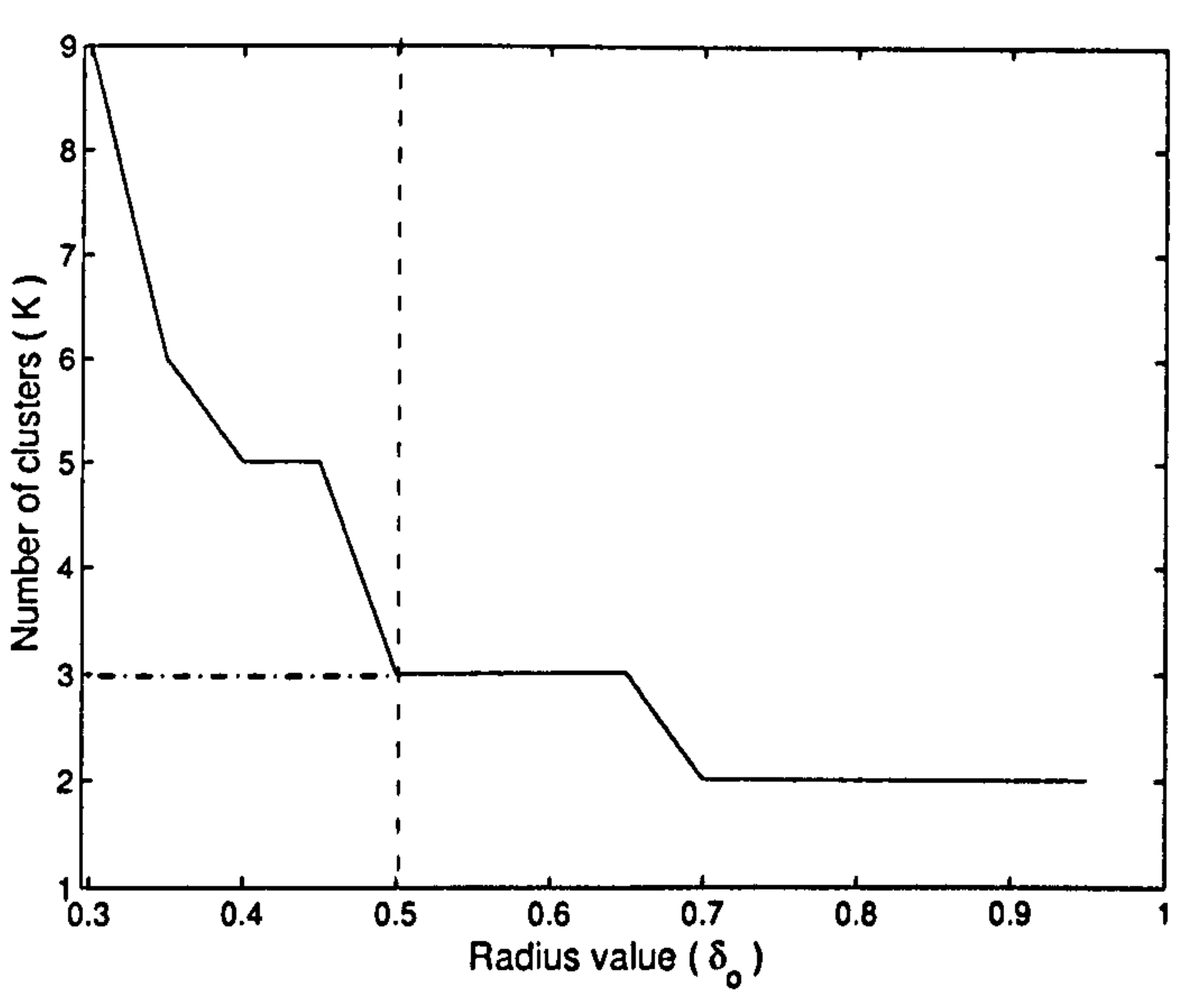
and treatment.

Similarly for liver cancer dataset as shown in Fig. 6.10, the number of clusters corresponding to the "knee" points for which I and CH validation indices are maximised and XB and DB validation indices are minimised, is now $K = 2$, which coincides with the true number of clusters. Finally, the obtained results indicate that the use of SOON algorithm along with a combination of validity indices significantly support genome expression analysis for biomedical knowledge discovery, and may help in identification of new tumor classes.

For purposes of comparison, the clustering results obtained by SOON are compared with the clustering results obtained by Hierarchical clustering (which is used in [110, 114]), SOM, and K-means clustering algorithms. The number of clusters K produced by the radius parameter δ_o using SOON clustering algorithm is used to partition the dataset by the other clustering algorithms that are considered in this work. Therefore, one can get fair comparison. Table 6.4 shows the evaluation of clustering results obtained by Hierarchical, SOM, K-means, and SOON clustering algorithms on lymphoma cancer data at different

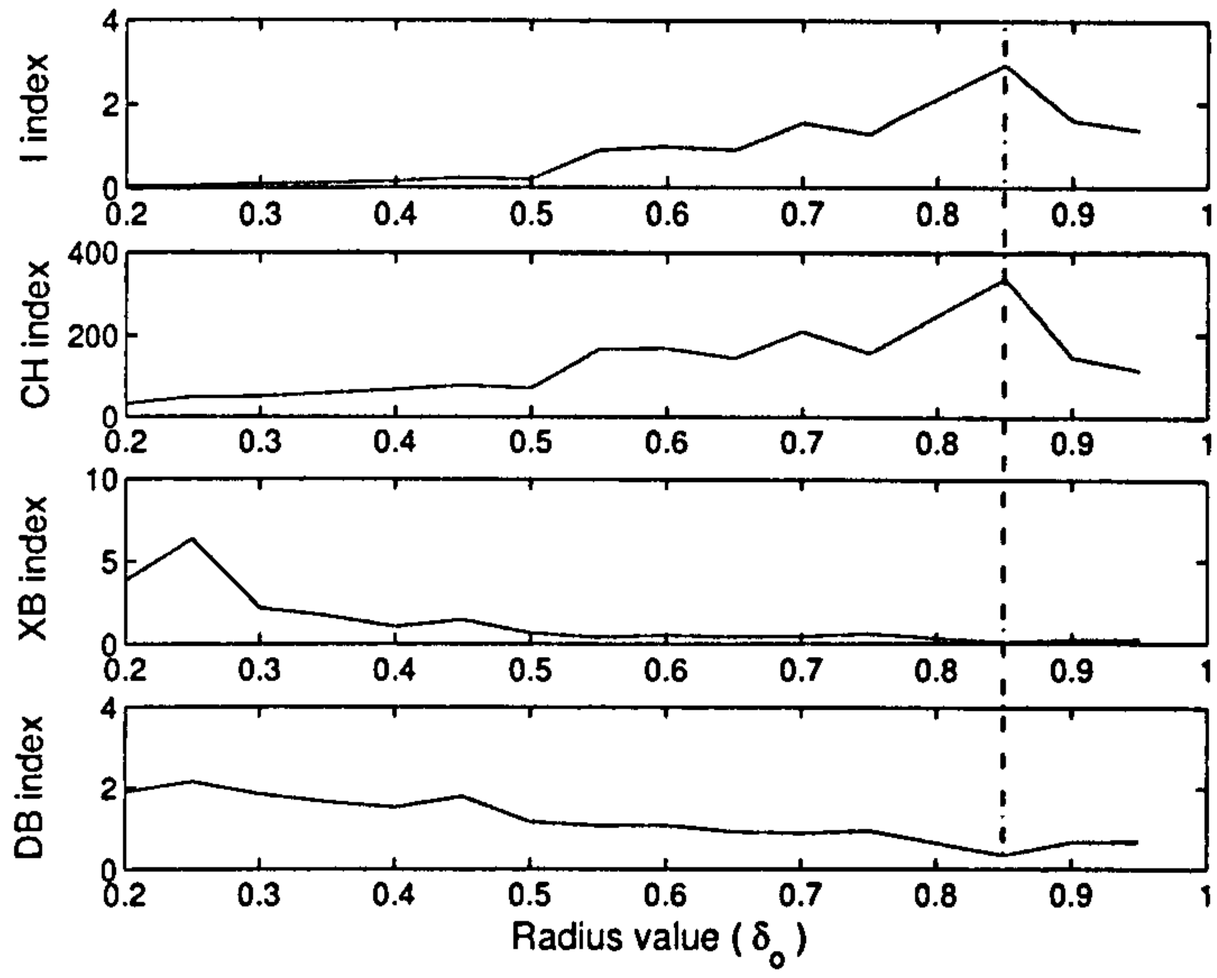


(a)

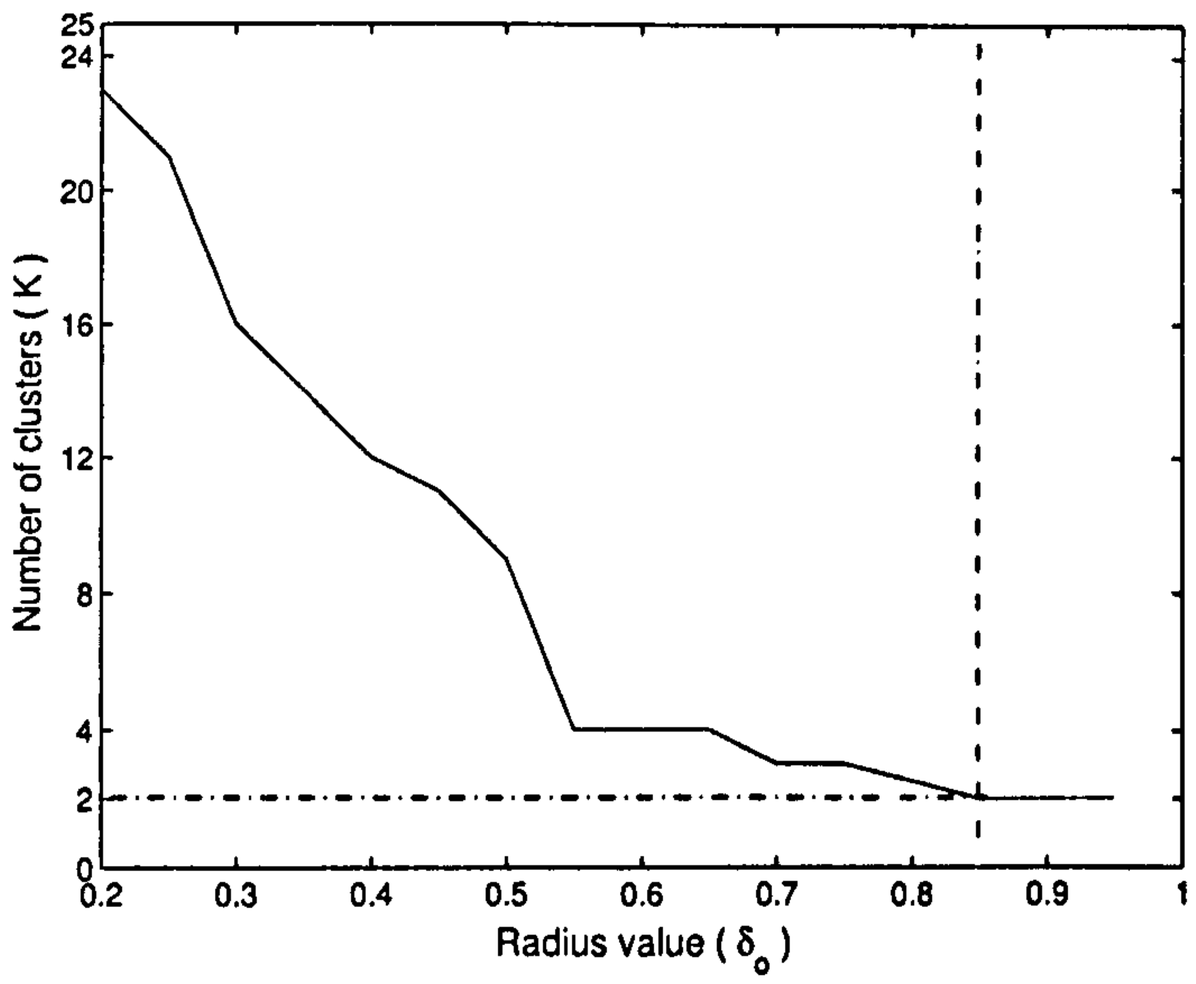


(b)

Figure 6.9: (a) Different index measures for a range of δ_o values on lymphoma cancer dataset, and (b) the number of clusters versus the radius δ_o.



(a)



(b)

Figure 6.10: (a) Different index measures for a range of δ_o values on liver cancer dataset, and (b) the number of clusters versus the radius δ_o.

Table 6.5: Comparisons with other clustering algorithms for liver cancer data.

Number of clusters	Validity index	Hierarchical	SOM	K-means	SOON
K = 2	*I* [59]	**1.95**	**1.59**	**1.60**	**2.93**
	CH [58]	**300.01**	**116.94**	**117.81**	**337.05**
	XB [64]	**0.16**	**0.20**	**0.19**	**0.075**
	DB [56]	**0.46**	**0.72**	**0.72**	**0.34**
K = 3	*I*	1.52	1.02	1.24	1.53
	CH	207.56	71.65	77.51	209.53
	XB	0.46	0.60	0.47	0.45
	DB	0.92	1.19	1.05	0.88
K = 4	*I*	0.89	0.69	0.82	0.97
	CH	153.29	54.32	57.29	167.53
	XB	0.55	0.71	0.65	0.51
	DB	1.52	2.06	1.93	1.08
K = 9	*I*	*0.16*	0.09	0.10	0.19
	CH	68.81	*42.47*	*43.19*	*71.64*
	XB	0.72	0.96	0.89	0.66
	DB	1.65	2.41	1.98	1.15
K = 12	*I*	0.13	0.07	0.08	0.15
	CH	60.84	35.91	38.01	66.73
	XB	1.20	2.26	1.74	1.06
	DB	1.68	2.48	2.09	1.52

number of clusters using different validation indices. As shown, the best values of validity indices for Hierarchical, SOM, and K-means clustering results estimate that two clusters are the best fit to the data, while the best values of validity indices for SOON clustering results indicate that three clusters are the best fit number of clusters which actually concurs with the true number of clusters. At $K = 3$, the true number of clusters, the validity indices values of SOON algorithm are better than the values obtained by the other clustering algorithms (i.e., higher values of *I* and *CH* indices, and lower values of *XB* and *DB* indices). These indicate the ability of SOON algorithm to discover the relevant clusters and it also achieves clustering of better quality (i.e., more compact clusters and yet wider separations between clusters). For liver cancer data as shown in Table 6.5, the number of clusters corresponding to the best values of validity indices for the clustering algorithms that are considered in this work are two clusters which are the best fit to the data. By comparing the validation indices values at $K = 2$ for all clustering algorithms, it is clearly seen that SOON algorithm achieves better validation values, i.e., better clustering quality.

6.3 Integrated Clustering Performance Measure

In the previous section, a combination of different validation methods have been used to assess clustering results obtained by SOON algorithm. This gives a novel idea for constructing a validity measure based on a combination of various validity indices to measure the quality of clustering results. Similarly, the stability of clustering results can be examined by constructing a stability measure based on a combination of various stability indices. In this section, a novel integrated clustering performance measure (*ICPM*) for assessing the reliability of results from a clustering algorithm used for analysing microarray data is proposed. The proposed *ICPM* is derived from these validity and stability measures which will be defined in this section. This *ICPM* can provide a guide for choosing clustering algorithms that have the ability to extract useful biological information from a given dataset.

6.3.1 ICPM - basis and definitions

For a reliable evaluation, two different kinds of measures, which are constructed based on a combination of various approaches, are used to measure the robustness and stability of a clustering algorithm. The first is the validity measure which is used for determining how well an algorithm works with a given set of parameter values. The second is the stability measure which is used for studying the consistency of clustering results. The proposed *ICPM* is derived from these two measures.

In the demonstration of the *ICPM*, two different approaches are used in this study:

- *Sub-sampling*: The idea is based on randomly selecting samples from a dataset with a sampling ratio S_R (a proportion of data points sampled) where $S_R < 1$. In this study, $S_R = 0.8$ is used. Therefore, one can get r datasets, each of which is sampled from the original dataset.

- *Leave-one-out*: The idea is based on deleting the observation (feature or attribute) at i-th position of either the expression profile vectors (in case of gene clustering) or the expressed genes (in case of sample clustering), i.e., for r observations, one can

get r datasets.

Then, a series of experiments are carried out with each of these r datasets to produce K clusters and for each value of K ($K = 2, \ldots, K_{max}$), i.e., one can get r sets of clustering results for every value of K clusters. In the following, the construction of a validity measure in Section 6.3.1.1 will be discussed, followed by the construction of a stability measure in Section 6.3.1.2, and finally the *ICPM* in Section 6.3.1.3.

6.3.1.1 Validity measure

In this study, a combination of different validation methods are used to define the validity measure where it is useful not to rely on one validation method, but to apply a variety of approaches. Therefore a combination of different validation methods can be successfully used for estimating the number of clusters and measure the quality of clustering results. In this study, eight validity indices are used in the construction of a validity measure to assess the clustering results which are obtained by clustering algorithms, namely, *DB* index [56], *DI* index [57], *CH* index [58], *I* index [59], *XB* index [64], *GI* index [102], *CS* index [116], and *Silhouette* index [117]. These indices have shown to be fairly robust for the predication of the optimal number of clusters [102, 103, 104, 105, 106].

Let nv be the considered number of validity indices; in this study nv is equal to eight. The overall validity measure is composed of the above nv (= 8) validity indices and is proposed as follows:

Suppose a clustering algorithm is applied to r datasets to produce K clusters, i.e., one obtains r sets of clustering results for a particular value of K. This is then repeated for every value of K ($K = 2, \ldots, K_{max}$). Every validity index can be applied to each of these r sets of clustering results. Let v_{ijKm} be the validation signature obtained for the clustering result generated by the i-th clustering algorithm on the j-th dataset ($j = 1, \ldots, r$) to produce K clusters ($K = 2, \ldots, K_{max}$) using the m-th validity index ($m = 1, \ldots, nv$). Let us further define

$$V_{iKm} = \frac{1}{r} \sum_{j=1}^{r} v_{ijKm} \tag{6.7}$$

where V_{iKm} is the average value of the m-th validity index when applied on r sets of clustering results from i-th clustering algorithm at K clusters.

In order to compare different clustering algorithms, the validation signature of each algorithm (at K clusters) is normalised by the maximum validation index value obtained from clustering algorithms as in Eq. 6.8.

$$V_{iKm}\,|_{norm} = \left\{ \frac{V_{iKm}}{\max\limits_{2 \le K \le K_{max}} (v_{ijKm},\, for\, j = 1, ..., r\,,\, i = 1, \ldots, nc)} \right\} \tag{6.8}$$

$$V_{iKm}\,|_{norm} = \{V_{iK1}\,|_{norm},\, V_{iK2}\,|_{norm},\, ...,\, V_{iK8}\,|_{norm}\} \tag{6.9}$$

where $0 \le V_{iKm}\,|_{norm} \le 1$, and nc is the number of clustering algorithms. Thus the $V_{iKm}\,|_{norm}$ is the normalised value of the m-th validity index, averaged over r sets of clustering results obtained from i-th clustering algorithm for K clusters.

The proximity of the value of the normalised clustering validity index of an algorithm (i) at K clusters to the ideal value is measured as follows:

$$CV_{iK} = \sqrt{\sum_{m=1}^{nv} (V_{ideal,m} - V_{iKm}\,|_{norm})^2} \tag{6.10}$$

where $V_{ideal,m}$ is the ideal normalised value of validity index m. In case of "I" index [59], $V_{ideal} = 1$, while $V_{ideal} = 0$ for "GI" index [102]. Hence, the overall *validity* measure for K clusters obtained from the i-th clustering algorithm is defined as:

$$Q_V(i,\, K) = 1 - \frac{CV_{iK}}{max\,(CV_{iK},\, for\, K = 2, ..., K_{max}\,,\, i = 1, \ldots, nc)} \tag{6.11}$$

where $0 \le Q_V(i,\, K) \le 1$. A higher value of $Q_V(i,\, K)$ indicates better clustering quality. The proposed validity measure, $Q_V(i,\, K)$ for K clusters (Eq. 6.11), is an integration of eight validity indices to assess clustering results which are obtained by clustering algorithms.

6.3.1.2 Stability measure

The stability of clustering results is evaluated using a combination of stability indices. Let ns be the considered number of stability indices; in this study ns is equal to six. Four of these are derived from statistics and similarity measure of partitions [2, 65]. The other two stability indices represent repeatability measure [86] and bootstrapping method [67].

The overall *stability* measure for K clusters obtained from i-th clustering algorithm can be constructed in a similar way as the overall *validity* measure in Section 6.3.1.1. Hence, the overall *stability* measure for K clusters obtained from the i-th clustering algorithm, using $ns(= 6)$ stability indices, is defined as:

$$Q_S(i,\, K) = 1 - \frac{CS_{iK}}{max\,(CS_{iK},\, for\, K\, = 2, ..., K_{max}\,,\, i = 1, \ldots, nc)} \tag{6.12}$$

where the proximity of the clustering stability of an algorithm (i) at K clusters is measured from:

$$CS_{iK} = \sqrt{\sum_{m=1}^{ns} (S_{ideal,m} - S_{iKm}\,|_{norm})^2} \tag{6.13}$$

where $S_{ideal,m}$ is the ideal normalised value of the stability index m and the $S_{iKm}\,|_{norm}$ is the normalised value of the m-th stability index, based on r sets of clustering results obtained from i-th clustering algorithm for K clusters. Note that, $0 \leq Q_S(i,\, K) \leq 1$. A larger value of $Q_S(i,\, K)$ indicates better stability. Thus, a good clustering algorithm is the one that can achieve higher values for many possible sets of parameter values. The proposed stability measure, $Q_S(i,\, K)$ for K clusters (Eq. 6.12), is an integration of six stability indices to assess the clustering results which are obtained by clustering algorithms.

6.3.1.3 Integrated clustering performance measure (ICPM)

The proposed *ICPM* provides an indication of the reliability of results obtained from a clustering algorithm. This is derived from the overall stability measure and the overall validity measure described above. Let $Q_S(i,\, K)$ and $Q_V(i,\, K)$ be the stability and validity values

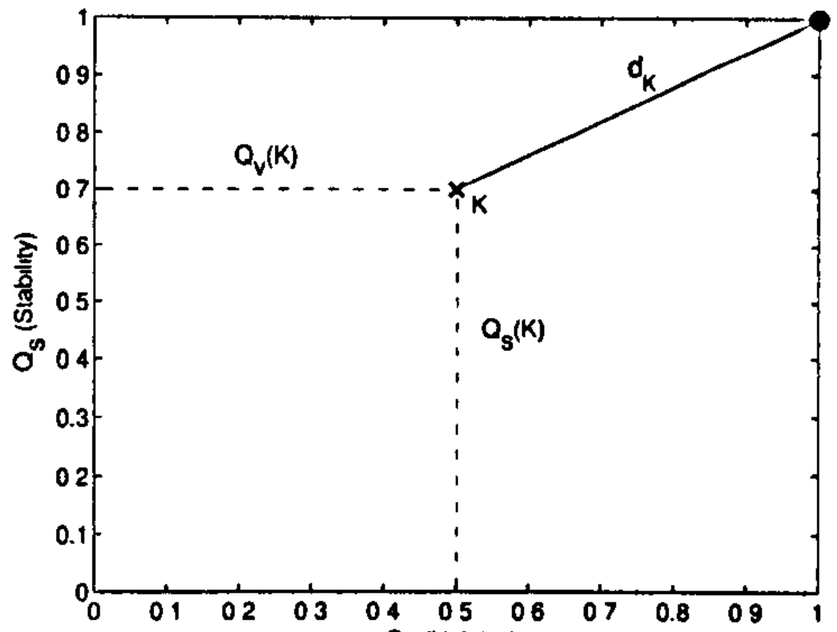


Figure 6.11: Integrated clustering performance measure - a pictorial representation.

corresponding to the K partitions (clusters) obtained by the i-th clustering algorithm, where $Q_S(i, K)$ is based on ns stability indices and $Q_V(i, K)$ is based on nv validity indices. Now it is worth referring to Fig. 6.11, which plots Q_V versus Q_S. Each set of r experimental results obtained from the i-th clustering algorithm corresponding to each value of K clusters will be represented by one point on this plot. It should be remarked that the ideal performance, that may possibly be achieved, is represented by the point with $Q_V = 1$ and $Q_S = 1$ (i.e. the top right-hand corner of the plot). Any point further away from this corner represents lower performance. Let $d_{i,K}$ be the distance of such a point from the ideal position, where $d_{i,K} = \sqrt{\alpha_V(1 - Q_V(i, K))^2 + \alpha_S(1 - Q_S(i, K))^2}$, $d_{i,K} \geq 0$, α_V and α_S are weighting parameters to represent the importance of Q_S and Q_V in the evaluation procedure. In this study, $\alpha_V = \alpha_S = 1$. The smaller distance corresponds to higher performance. The integrated clustering performance measure is defined as:

$$ICPM = min(d^2_{i,K}, \, for \, K = 2, ..., K_{max}) \tag{6.14}$$

This takes into account of all points (corresponding to different partitions for different values of K) and ensures that smaller values of *ICPM* represents better clustering performance.

For the yeast cell cycle dataset, *Fig. 6.12 plots* the validity measure, $Q_V(i, K)$, and the stability measure, $Q_S(i, K)$, corresponding to K clusters for each of the nc clustering algorithms. Each point refers to a specific number of clusters (a particular value of K) and

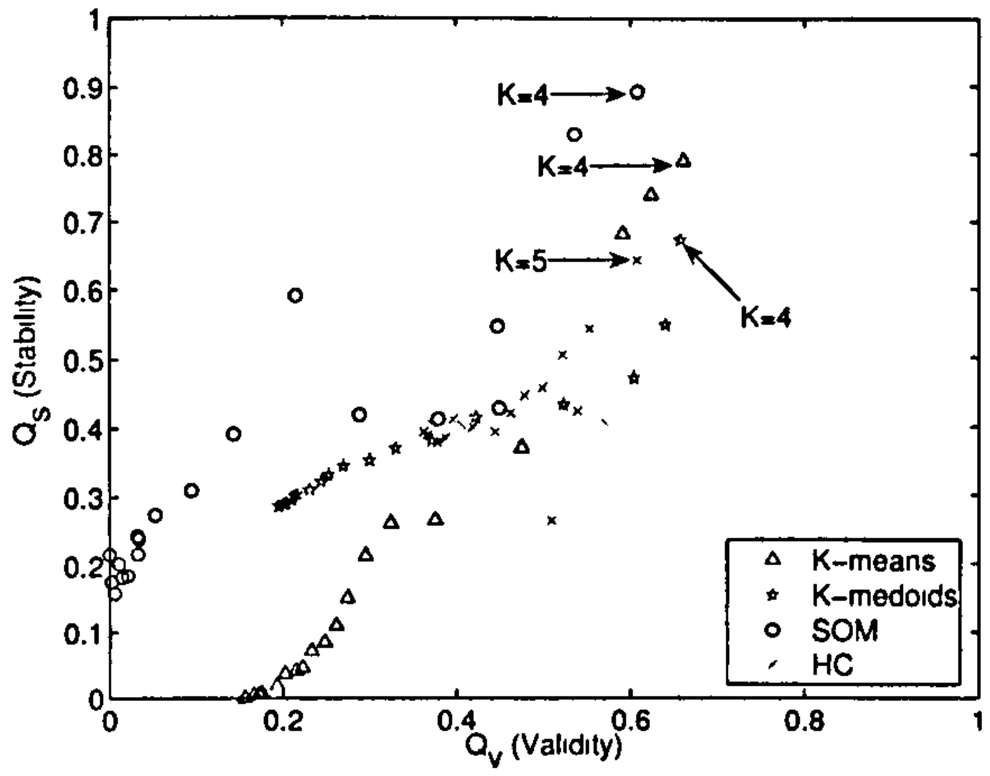


Figure 6.12: Relationship between validity and stability measures for yeast cell cycle dataset.

a specific clustering algorithm. The four sets of clustering results chosen by the *ICPM* - one from each of the four clustering algorithms - are marked on the plot by the corresponding values of K.

This measure can be used to obtain fair comparison between clustering algorithms from the two important points of view - the stability and validity - for the identical range of desired numbers of clusters. In should be noted that this *ICPM* can be extended to take into account other measures of clustering, e.g., the computational complexity, the effect of data size, number of dimensions, etc. Therefore, The general *ICPM* is proposed as follows:

$$ICPM = \sum_{j=1}^{c} \alpha_j (1 - Q_i)^2 \tag{6.15}$$

where c is the number of different measures, and $\alpha_j{}'s$ are weighting parameters of a measure j, $\alpha_j \in [0, 1]$.

6.3.2 Datasets

6.3.2.1 Yeast cell cycle dataset

The yeast cell cycle dataset [109] shows the fluctuation of expression levels of approximately 6,400 genes over approximately two cell cycles (17 time points). In the study of

Yeung *et al.* [103], only a subset of 384 genes was adopted and used for the purpose of validation and can be found at http://faculty.washington.edu/kayee/model/. The dataset is normalised as in Tamayo *et al.* [98] to zero mean and unit standard deviation.

6.3.2.2 Leukaemia dataset

The data contain 38 samples (27 acute lymphoblastic leukaemia (ALL) and 11 acute myeloid leukaemia (AML)) described by 7,129 probes from 6,817 human genes. In this study of [118], only 50 genes have suspected roles in this type of cancer. These data were obtained from a study published by Golub and co-workers [118] and can be found at http://sdmc.lit.org.sg/GEDatasets/Datasets.html#ALL-AML_Leukemia.

6.3.2.3 Lymphoma dataset

Distinct types of diffuse large B-cell lymphoma (DLBCL) is the most common subtype of non-Hodgkin's lymphoma. There are 47 samples, 24 of them are from the "germinal centre B-like" group while 23 are from the "activated B-like" group. Each sample is described by 4,026 genes. In the study of [110], only 100 genes, that best differentiate a germinal centre B-like from an activated B-like, are considered. This dataset can be found at http://llmpp.nih.gov/lymphoma/.

6.3.2.4 Rat CNS dataset

The rat CNS data was obtained by reverse transcription-coupled PCR to study the expression levels of 112 genes during rat central nervous system development [119] over nine time points. As suggested in [119], the raw data was normalised by the maximum expression level for each gene. The dataset was then augmented with slopes (differences between consecutive time points) to capture parallel trajectories of the time series data. This results in a dataset with 112 genes and 17 conditions. Wen *et al.* [119] categorised genes in the rat CNS dataset into four families using biological knowledge.

6.3.3 Clustering algorithms

In this study, four commonly used clustering algorithms are considered. These algorithms are popular in the biological community, particularly in the field of microarray analysis. These include hierarchical clustering algorithm with average linkage (HC), SOM, K-means, and K-medoids clustering algorithms [96, 97, 98, 120].

6.3.4 Experimental results

In this sub-section, comparisons of various clustering algorithms, which are applied on the aforementioned datasets, are presented. These comparisons are achieved using the proposed integrated clustering performance measure (*ICPM*) and two different approaches are used for reliable conclusions.

6.3.4.1 Sub-sampling approach

In this approach, from each dataset, random samples from the entire dataset are selected with a sampling ratio $S_R = 0.8$. By repeating this process r times, r datasets are obtained, each dataset is sampled from the entire dataset. Then, the clustering algorithms are applied to every sub-sampled dataset ($j = 1, \ldots, r$) to find K clusters ($K = 2, \ldots, 20$), i.e., one can get r clustering results for every K. In this study, $r = 100$ is used. It should be remembered that at the first stage of clustering, not all objects are clustered due to sub-sampling. In the following experiments, 20% of the samples, as $S_R = 0.8$, will need to be assigned to clusters. To address this issue, each unassigned sample is assigned to the most appropriate cluster that is consistent with the clustering algorithm. For example, in the case of K-means algorithm, each unassigned sample is assigned to the cluster with the closest centre.

Figures 6.13, 6.14, 6.15, and 6.16 present results corresponding to each of these four datasets and each of these figures contain two subplots each. The subplot on top shows the validity measure versus the number of chosen clusters for each of the four different clustering algorithms. For each of the clustering algorithms, the number of clusters corresponding to the "knee" point is an estimate of the true number of clusters in the underlying

dataset. Of course, the higher validity values for any given number of clusters corresponds to better robustness. The lower subplot shows the stability measure over the same number of clusters. As shown, the number of clusters corresponding to the "knee" points of the stability for the four clustering algorithms is an estimate of the true number of clusters in each of the underlying datasets, i.e. the estimated number of clusters is corresponding to the higher stability value. Clearly, a better clustering algorithm is one that achieves higher validity as well as stability values.

Tables 6.6, 6.7, 6.8 and 6.9 contain four values per clustering algorithm. The first is the expected value of the validity measure, averaged over different numbers of clusters K ($K = 2, \ldots K_{max}$). The second is the expected value of the stability measure, averaged over different numbers of clusters K ($K = 2, \ldots K_{max}$. The third corresponds to the *ICPM* value, while the fourth provides the value of K_{est}, the estimated number of clusters, corresponding to the *ICPM*.

For the yeast cell cycle dataset, as shown in Fig. 6.13, the validity and stability measures of K-means, K-medoids, and SOM algorithms select four ($K = 4$) as the best number of clusters underlying the dataset, which correspond to the main phases G1, S, G2 and M, as identified by Cho *et al.* [109], Spellman *et al.* [108], and Tamayo *et al.* [98], while the validity and stability measures of HC algorithm select five ($K = 5$) as the best number of clusters where the fifth cluster corresponds to late G1 phase [102]. Thus in this dataset both four and five clusters have been reported by the biological community. The design of HC clustering algorithm is based on a hierarchical decomposition of the data, which is represented by a tree structure that splits the data into small subsets until each subset consists of only one object. Each node of the tree represents a cluster of data. In case of K-means, K-medoids, and SOM clustering, optimisation criteria are used to minimise within-cluster discrepancies, and these reflect the choice of four clusters by validity and stability measures of K-means, K-medoids and SOM, as these algorithms favour compact clusters. K-means and SOM algorithms achieve lower *ICPM* values as recorded in Table 6.6. These algorithms give lower validity and stability values than HC and K-medoids clustering algorithms, where HC algorithm achieves higher validity as well as better stability

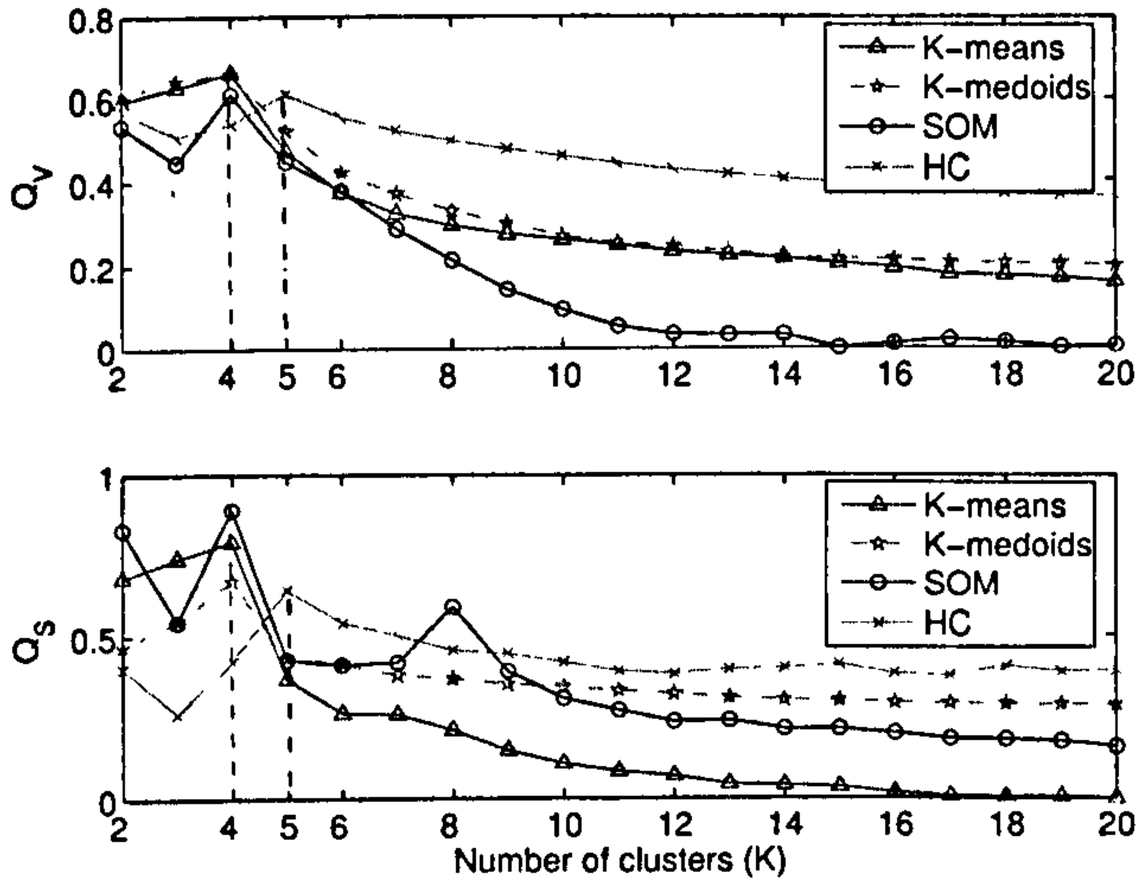


Figure 6.13: The validity (Q_V) and stability (Q_S) measures versus the number of clusters for yeast cell cycle dataset using sub-sampling approach.

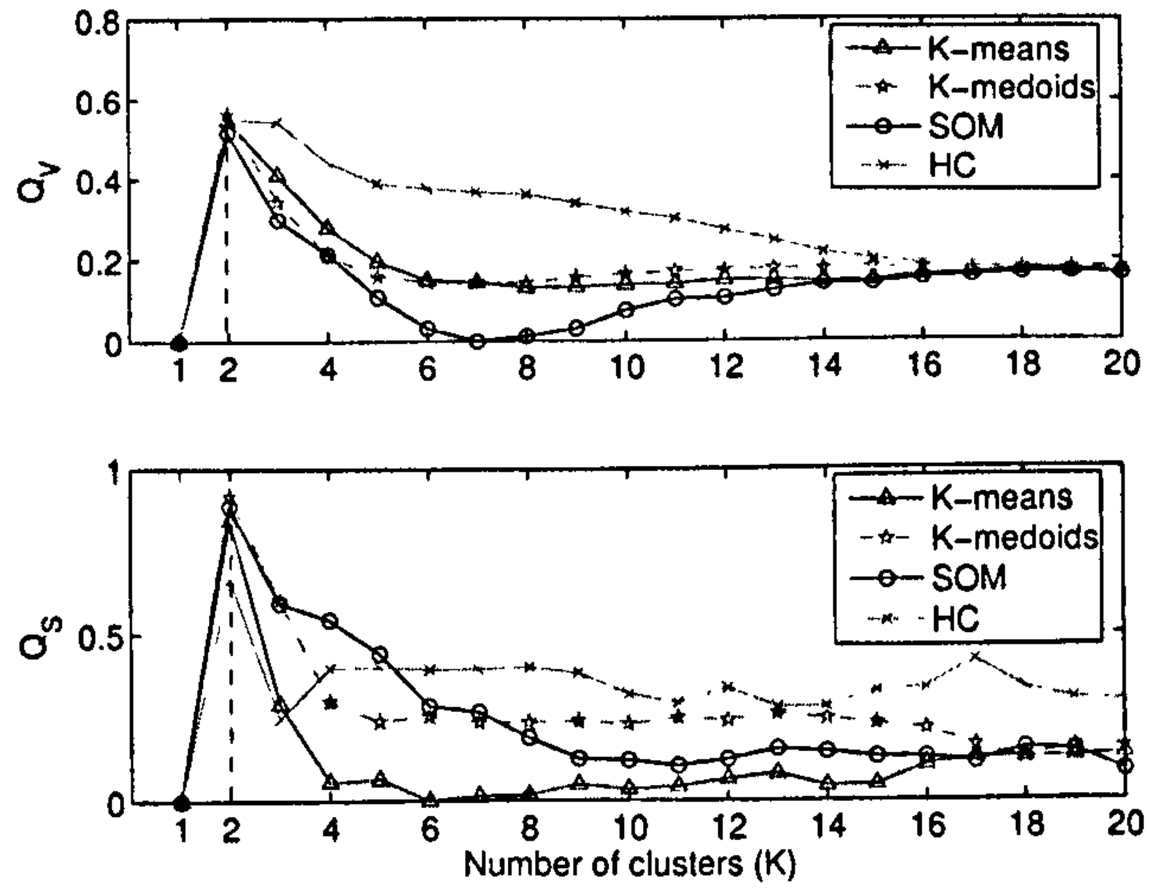


Figure 6.14: The validity (Q_V) and stability (Q_S) measures versus the number of clusters for leukaemia dataset using sub-sampling approach.

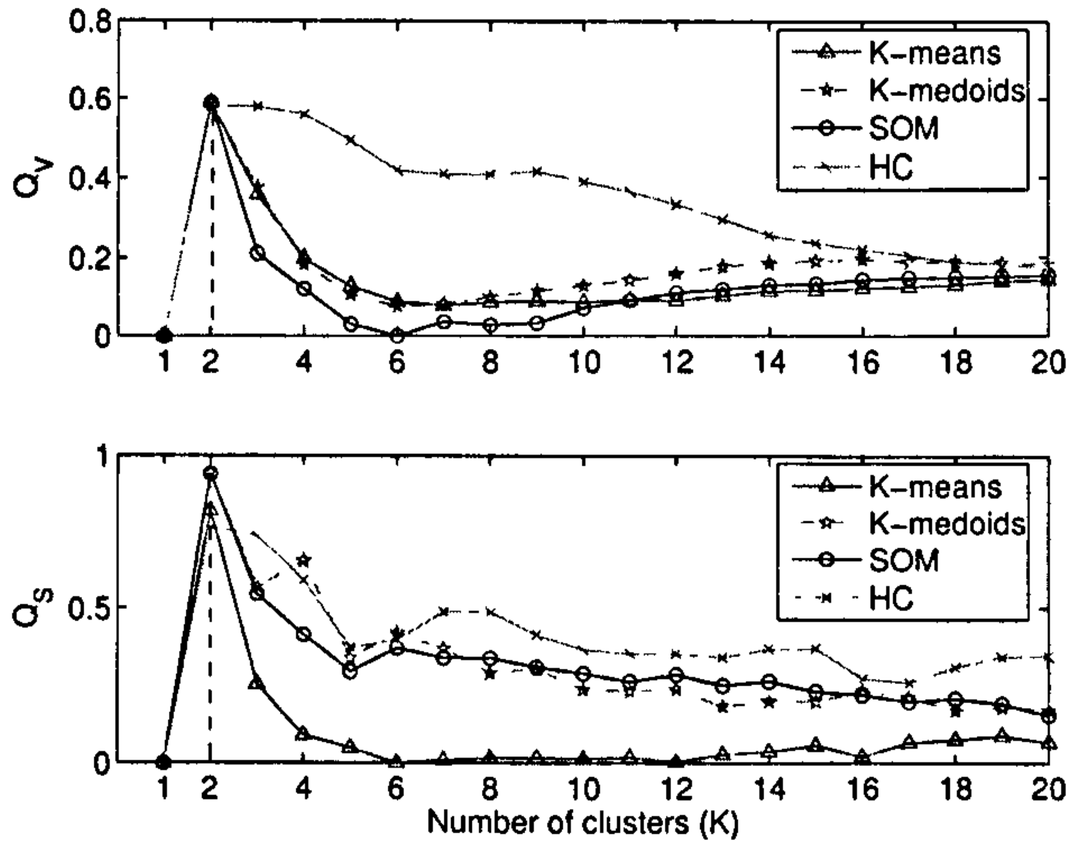


Figure 6.15: The validity (Q_V) and stability (Q_S) measures versus the number of clusters for lymphoma dataset using sub-sampling approach.

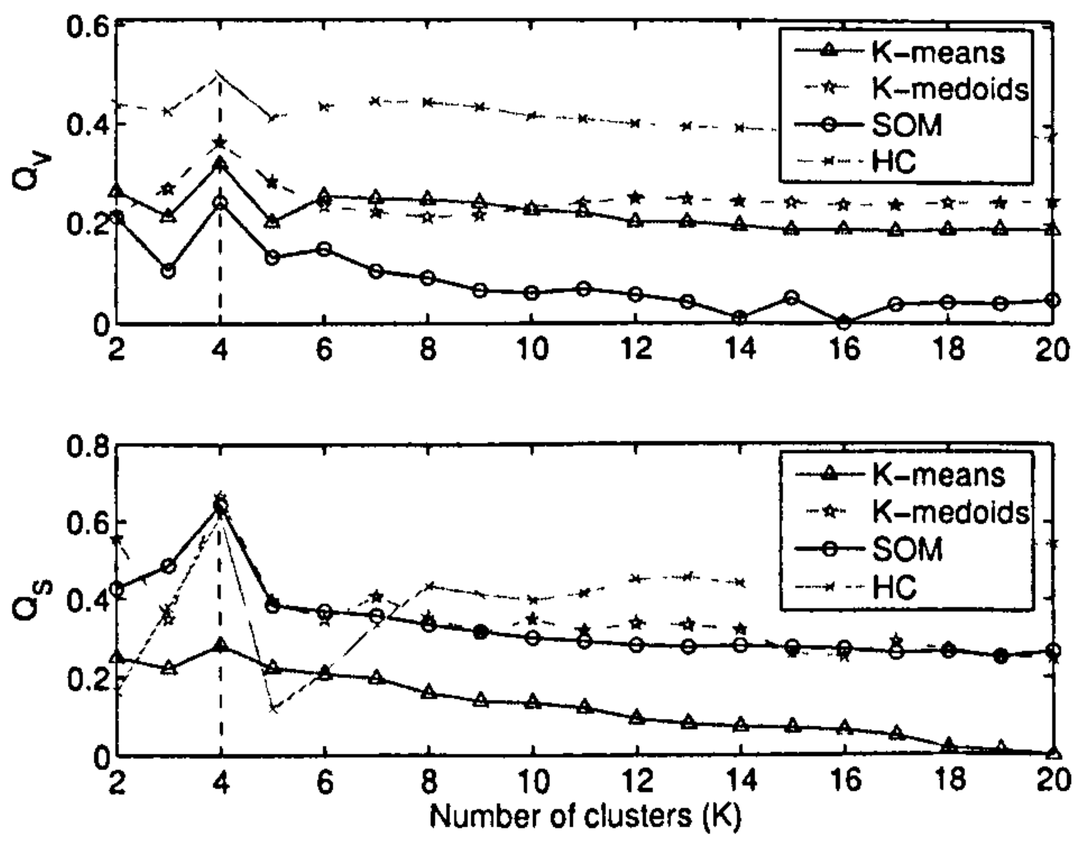


Figure 6.16: The validity (Q_V) and stability (Q_S) measures versus the number of clusters for rat CNS dataset using sub-sampling approach.

than the other algorithms. Overall, all four algorithms perform well.

For the leukaemia, lymphoma, and rat CNS datasets, as shown in figures 6.14, 6.15, and 6.16, the validity and stability measures of all clustering algorithms correctly estimate the true number of clusters in the underlying datasets. Corresponding results are presented in Tables 6.7, 6.8, and 6.9. In the case of leukaemia and lymphoma datasets all four algorithms offer similar results, while for the rat CNS dataset the HC algorithm offers the best results, followed by K-medoids and then SOM which is followed by the worst performing K-means.

Table 6.6: Performance evaluation results on yeast cell cycle dataset using sub-sampling approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.31	0.33	0.18	0.45
$< Stability >$	0.21	0.37	0.36	0.42
$ICPM$	0.16	0.23	0.17	0.28
K_{est}	4	4	4	5

Table 6.7: Performance evaluation results on leukaemia dataset using sub-sampling approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.19	0.20	0.14	0.31
$< Stability >$	0.12	0.28	0.25	0.36
$ICPM$	0.23	0.20	0.25	0.31
K_{est}	2	2	2	2

Table 6.8: Performance evaluation results on lymphoma dataset using sub-sampling approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.15	0.19	0.13	0.35
$< Stability >$	0.09	0.32	0.32	0.42
$ICPM$	0.20	0.17	0.17	0.23
K_{est}	2	2	2	2

Table 6.9: Performance evaluation results on rat CNS dataset using sub-sampling approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.23	0.25	0.08	0.41
$< Stability >$	0.13	0.35	0.33	0.41
$ICPM$	0.98	0.52	0.71	0.40
K_{est}	4	4	4	4

6.3.4.2 Leave-one-out approach

In this approach, the effect of features (observations or attributes) on the stability and quality of clustering results is examined by omitting the feature at i-th position of either the expression profile vectors (in case of gene clustering) or the expressed genes (in case of sample clustering), i.e., for r features, one can get r datasets. Then, the four clustering algorithms are applied to every dataset with K ($K = 2, \ldots, 20$), i.e. one can get r clustering results for every K.

By comparing the stability and validation measures obtained from leave-one-out approach with those achieved by sub-sampling approach, although the quality of the clustering results, which is represented by validity measure, does not have a significant change compared with sub-sampling approach, i.e., comparable results. There is a significant change in stability measures at different datasets, e.g., K-medoids achieves higher values at leukaemia, lymphoma, and rat CNS data in case of leave-one-out approach compared with values obtained from sub-sampling approach.

Finally, by examining the *ICPM* values as demonstrated in Tables 6.10, 6.11, 6.12, and 6.13, HC algorithm achieves on the whole higher average validity as well as better stability, and K-medoids algorithms is chosen as a second choice with better *ICPM* values. Again, the proposed *ICPM* of all four clustering algorithms on the four microarray datasets verifies the correct estimation of the number of clusters.

Table 6.10: Performance evaluation results on yeast cell cycle dataset using leave-one-out approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.32	0.35	0.19	0.48
$< Stability >$	0.23	0.39	0.29	0.46
$ICPM$	0.16	0.14	0.25	0.29
K_{est}	4	4	4	5

Table 6.11: Performance evaluation results on leukaemia dataset using leave-one-out approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.22	0.25	0.16	0.35
$< Stability >$	0.10	0.64	0.41	0.67
$ICPM$	0.22	0.17	0.23	0.17
K_{est}	2	2	2	2

Table 6.12: Performance evaluation results on lymphoma dataset using leave-one-out approach approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.16	0.20	0.13	0.37
$< Stability >$	0.12	0.82	0.75	0.66
$ICPM$	*0.19*	*0.18*	*0.20*	0.24
K_{est}	2	2	2	2

Table 6.13: Performance evaluation results on rat CNS dataset using leave-one-out approach.

	K-means	K-medoids	SOM	HC
$< Validity >$	0.26	0.28	0.09	0.45
$< Stability >$	0.08	0.51	0.19	0.42
$ICPM$	1.09	0.45	0.62	0.45
K_{est}	4	4	4	4

6.3.5 Clustering accuracies

In this sub-section, accuracies of clustering results from two microarray cancer datasets - leukaemia and lymphoma datasets using sub-sampling approach are investigated. It has already been observed from Tables 6.7 and 6.8 that *ICPM* values of each of the four clustering algorithms estimate correctly the true number of clusters in the underlying datasets. From each of these datasets, two sets of 100 sub-sampled datasets have been generated using the previously described sub-sampling approach. Each of these 100 sub-sampled datasets have been clustered using four aforementioned clustering algorithms. Results obtained are averaged for clustering accuracies and presented in Table 6.14. In case of leukaemia dataset, K-means, K-medoids and SOM algorithms achieve similar average clustering accuracies of around 98%, while HC algorithm achieves clustering accuracy of 91% with a high standard deviation. In the case of lymphoma dataset, all four algorithms achieve similar average clustering accuracies of around 98%, though HC has a high standard deviation. It should be noted that there is no supervised learning or training involved here but only clustering with the aid of *ICPM*. In the absence of any training, the clustering accuracy of 98% for the lymphoma dataset is remarkable. For the purpose of comparisons, in the case of lymphoma dataset, classification performances of 84.6% using KNN classifier [121], 95.0 % using either nearest neighbour classifier or diagonal linear discriminant analysis [122], and

Table 6.14: Clustering accuracies based on 100 runs using sub-sampling approach on two microarray cancer datasets.

	Leukaemia data	Lymphoma data
K-means (%)	98.4 ± 2.8	98.7 ± 1.1
K-medoids (%)	98.1 ± 1.2	99.3 ± 1.0
SOM (%)	97.3 ± 1.8	98.5 ± 0.9
HC (%)	91.0 ± 13.4	97.0 ± 4.5

98.1% using logistic discrimination [123]. In the case of leukaemia dataset, classification performance of 94.7% using cross validation sets was achieved [118].

6.4 Summary

In this chapter, the efficacy of SOON algorithm, that does not require the knowledge of the number of clusters, in analysing biological datasets has been examined. As shown, it is possible to change clustering behaviour to show the desired characteristics (small, tight clusters or large, loose clusters) through the choice of the values of α and δ_0. The useful behaviour of the algorithm on non-separable data has been verified with a series of validation sets with varying values of SNR. The useful properties of the clustering algorithm were also demonstrated using microarray datasets along with different distance measures. Experiments on microarray yeast data has shown the ability the SOON algorithm to discover new patterns which are not discoverd by the SOM algorithm. Furthermore, the assessment of SOON clustering results using a combination of validation methods indicate the ability of SOON to discover the clusters of a dataset and also achieve clustering of better quality.

Finally, a novel integrated clustering performance measure (*ICPM*) for assessing the reliability of results from a clustering algorithm has been proposed. This is composed of a validity measure, which is constructed in this study out of eight validity indices, and a stability measure, which is constructed in this study out of six stability indices. The proposed *ICPM* has been tested using different types of microarray data. Experimental results indicate that the integration of clustering performance measures with various approaches to measure the stability and quality of clustering results can be used for evaluating different clustering algorithms and examining shortfalls of such algorithms in a robust way.

As demonstrated, the proposed *ICPM* provides a guide for examining the suitability of an algorithm for clustering microarray data. In these experiments on different microarray case studies, the proposed *ICPM* estimates correctly the true number of clusters in the underlying datasets. The number of validity indices, the number of stability indices can be changed as well as the proposed *ICPM* can be extended naturally to include other desirable measures than validity and stability.

Chapter 7

Conclusions and Future Work

7.1 Summary and Conclusions

Clustering is a common tool for data analysis. Of course there exist many clustering algorithms already but there is a demand for a guideline for choosing the proper clustering algorithm for a particular dataset, new algorithms for achieving clustering with no a priori knowledge of the data, and efficient methods for handling the rapid growth of the data. This thesis has proposed possible solutions for these challenges.

Chapter 3 has proposed new measures (*CPMs*) for the robustness and reliability of clustering algorithms. These *CPMs* are used to evaluate clustering algorithms that have a structural bias to a certain type of data distribution as well as those that have no such bias. As demonstrated, the effect of initial and random guesses on the clustering algorithms that have initialisation dependency is tested and evaluated using different types of real-world data and synthetic data which contain well-separated datasets, as well as overlapped datasets. These measures have been applied on sub-sampled datasets and datasets with less separated clusters. Therefore, one can use the proposed *CPMs* to evaluate the clustering algorithms that have a unique solution for a given set of parameter values with no initialisation dependency (e.g. hierarchical clustering algorithm). Additionally, these measures can be used for evaluating the performance of different clustering algorithms and examining shortfalls of such algorithms.

Chapter 4 has proposed a new clustering algorithm, *RACAL*, which does not require the knowledge of the number of clusters. As demonstrated, it is possible to control the behaviour of the cluster membership (small or large) through a radius parameter δ_o. Moreover, *RACAL* is augmented with a reliable validation index to produce the best results for the given δ_o values. Four different types of real-world data have been used to demonstrate that the proposed algorithm scales well with the size and dimension of the dataset. The clustering performance measures, *CPMs*, indicate that *RACAL* algorithm possesses higher robustness as well as offers better repeatability (stability) compared with the other mentioned clustering algorithms. In addition, an adaptive partial supervision strategy has been proposed for *RACAL* to make it act as a classifier. Experimental results obtained by *RACAL-PS* show that it achieves higher classification performance than other classification methods. Additionally, a parallel version of *RACAL* (*P-RACAL*) has been proposed. As demonstrated, *P-RACAL* is scalable in terms of speedup and scaleup, which gives the ability to handle large datasets of high dimensions in a reasonable time.

Chapter 5 has proposed a novel clustering algorithm, *NNCA*, which achieves clustering without any control of cluster sizes. In addition, *NNCA* has been augmented with a partial supervision strategy to act as a classifier. Comparisons with other methods have indicated the robustness of the proposed method in classification. Experimental results show that the *NNCA-PS* offers better classification accuracies compared with other classifiers. Additionally, a parallel version of *NNCA* (*P-NNCA*) has been proposed. As demonstrated, *P-NNCA* is scalable in terms of speedup and scaleup, which gives the ability to handle large datasets of high dimensions in a reasonable time.

Chapter 6 has examined the efficacy of *SOON* algorithm, that does not require the knowledge of the number of clusters, in analysing biological datasets. As shown, it is possible to change clustering behaviour to show the desired characteristics (small, tight clusters, or large, loose clusters) through the choice of the values of α and δ_0. The useful behaviour of the algorithm on non-separable data has been verified with a series of validation sets with varying values of SNR. The useful properties of the clustering algorithm were also demonstrated using microarray datasets. Furthermore, the assessment of *SOON*

clustering results using a combination of validation methods indicate the ability of *SOON* to discover the clusters of a dataset and also achieve clustering of better quality.

Finally, a novel integrated clustering performance measure (*ICPM*) for assessing the reliability of results from a clustering algorithm used for analysing microarray data has been proposed. This is composed of a validity measure, which is constructed in this study out of eight validity indices, and a stability measure, which is constructed in this study out of six stability indices. The proposed *ICPM* has been tested using different types of microarray data. Experimental results indicate that the integration of clustering performance measures with various approaches to measure the stability and quality of clustering results can be used for evaluating different clustering algorithms and examining shortfalls of such algorithms in a robust way. As demonstrated, the proposed *ICPM* provides a guide for examining the suitability of an algorithm for clustering microarray data. In the experiments on different microarray case studies, the proposed *ICPM* estimates correctly the true number of clusters in the underlying datasets. The number of validity indices, the number of stability indices can be changed as well as the proposed *ICPM* can be extended naturally to include other desirable measures than validity and stability.

7.2 The Road Ahead

The following is a list of possible avenues for the continuation of this work:

- Every clustering algorithm has certain advantages over others, and there is no clustering algorithm that can be universally used to solve all problems. Therefore, the development of an integrated clustering algorithm based on a combination of various clustering approaches can give better and meaningful clustering results. This requires a structure phase test to examine the data tendency. Then, a selection phase selects the best fit clustering criteria to the examined data structure. Of course, one might think of the high computational complexity, but it is preferable to take into consideration the recent advances in computer networking, data storage technologies, and parallel computation in reducing the computational resources as well as improving the clustering performance and accuracy.

- Extend the use of *NNCA* and *RACAL* with partial supervisions for microarray data classification.

- Incorporate the constructed validity measure, Q_V, with a clustering criterion for achieving automatic data clustering.

- As there are a lot of applications which would benefit from efficient parallel clustering algorithms, further studies are needed to unearth the potential of these emerging algorithms.

Appendix A

Validity indices

Validity indices are used for judging the relative merits of clustering structures in a quantitive manner, and selecting appropriate parameter settings. The following are seven validity indices used in this thesis.

"*GI*" Index: This index [102] measures the geometrical feature of the data. Its principle is to measure the value of the squared total length of the eigen-axes of the data with respect to the between-cluster separation. This index is defined as:

$$GI = max_{1 \leq k \leq K} \left\{ \frac{\left(2 \sum_{j=1}^{d} \sqrt{\lambda_{jk}}\right)^2}{min_{1 \leq q \leq K} \left(\|z_k - z_q\|_2\right)} \right\} \tag{A.0.1}$$

where K is the number of clusters, d is the number of dimensions, $\lambda_{jk}(k = 1, \ldots, K)$ are eigenvalues of the covariance matrix, z_k and z_q are cluster centres for i-th and q-th clusters respectively. If the value of GI index reaches the minimum, the clustering result is an optimum solution. The ideal value for the GI index is 0.

"*CS*" Index: This index [116] is a function of the ratio of the sum of the within-cluster scatter to between-cluster separation. This index is defined as:

$$CS = \frac{\sum_{i=1}^{K} \left\{ \frac{1}{|A_i|} \sum_{x_j \in A_i} max_{x_k \in A_i} \{d(x_j, x_k)\} \right\}}{\sum_{i=1}^{K} \{min_{j \in K, j \neq i} \{d(z_i, z_j)\}\}} \tag{A.0.2}$$

where d is a distance function, A_i is the set of data points assigned to i-th cluster, $|A_i|$ is the number of data points in A_i, z_i and z_i represent the i-th and j-th cluster centres

respectively. The smallest value of CS index indicates a valid optimal partition. The ideal value for the CS index is 0.

***"CH"* Index:** Calinski Harabasz index [58], for n data points and K clusters is computed as $\frac{trace\,(B)/(K-1)}{trace\,(W)/(n-K)}$, where B and W are the between-cluster separation and within-cluster scatter matrices. The maximum hierarchy level is used to indicate the correct number of partitions in the data. The trace of the between cluster scatter matrix B can be calculated as $trace\ (B) = \sum_{k=1}^{K} n_k \left\|z_k - z\right\|^2$, where n_k is the number of points in cluster k and z is the centroid of the entire dataset. The trace of the within cluster scatter matrix W can be calculated as $trace\ (W) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} \left\|x_i - z_k\right\|^2$. Therefore, the CH index is defined as:

$$CH = \left[\frac{\sum_{k=1}^{K} n_k \left\|z_k - z\right\|^2}{K-1}\right] / \left[\frac{\sum_{k=1}^{K} \sum_{i=1}^{n_k} \left\|x_i - z_k\right\|^2}{n-K}\right] \qquad \text{(A.0.3)}$$

The objective is to maximise the CH index for achieving proper clustering. The ideal normalised value for the CH index is 1.

***"DI"* Index:** Dunn index [57] is defined as the ratio of the between-cluster separation to the within-cluster scatter. The DI is defined as:

$$DI = \frac{1}{n} min_{1\leq i\leq K} \left\{ min_{1\leq j\leq K} \left\{ \frac{\delta(z_i, z_j)}{max_{1\leq k\leq K} \{\Delta(z_k)\}} \right\} \right\} \qquad \text{(A.0.4)}$$

where:

$\delta(z_i, z_j) = min \left\{ \left\|x_i - x_j\right\|_2 \ | x_i \in z_i,\ x_j \in z_j \right\}$, $\Delta(z_k) = max \left\{ \left\|x_i - x_j\right\|_2 \ | x_i,\ x_j \in z_j \right\}$.

Large values of DI represent good clustering results and the value of K that maximises DI corresponds to the optimal number of clusters. The ideal normalised value for the DI index is 1.

***"Silhouette"* Index:** For a given cluster, $X_j (j = 1, \ldots, K)$, this index [117] assigns each data point (i) of cluster X_j a quality index, $s(i)$ $(i = 1, \ldots, n)$, known as the *Silhouette width*. The Silhouette width is a confidence indicator on the membership of the i-th data point in cluster X_j, and it is defined as:

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

where $a(i)$ is the average distance between the i-th data point and all points included in X_j, and $b(i)$ is the minimum average distance between the i-th data point and all points clustered in $X_k(k = 1, \ldots, K; k \neq j)$. From this formula, it follows that $-1 \leq s(i) \leq 1$. When $s(i)$ is close to 1, one may infer that i-th data point has been well-clustered. When $s(i)$ is close to zero, it suggests that i-th data point could also be assigned to the nearest neighbouring cluster. If $s(i)$ is close to -1, one may argue that such a data point has been misclassified. Thus for a given cluster $X_j(j = 1, \ldots, K)$, it is possible to calculate the Silhouette S_j, which characterises the heterogeneity and isolation properties of such a cluster:

$$S_j = \frac{1}{n} \sum_{i=1}^{n} s(i)$$

where m is the number of data points in X_j.

Therefore, for any partition $U \leftrightarrow X : X_1 \cup \ldots X_i \cup \ldots X_K$, a Global Silhouette index, GS, can be used as an effective validity index for U.

$$GS = \frac{1}{K} \sum_{j=1}^{K} S_j \tag{A.0.5}$$

The most appropriate number of clusters corresponds to the maximum value of GS index. The ideal normalised value for the Silhouette index is 1.

***"XB"* Index:** Xie Beni index [64] is defined as the ratio between the compactness π of the fuzzy K-partition of a dataset to the minimum separation s of the clusters. Here π and s can be written as $\pi = \sum_{k=1}^{K} \sum_{i=1}^{n_k} u_{kj}^2 \|x_i - z_k\|^2$and $s = min_{i \neq j} \|z_i - z_j\|^2$. The XB index is then defined as follows:

$$XB = \frac{\pi}{n * s} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{n_k} u_{kj}^2 \|x_i - z_k\|^2}{n * min_{i \neq j} \|z_i - z_j\|^2} \tag{A.0.6}$$

Note that when the partitioning is compact, value of π should be low while s should be high,

thereby yielding lower values of the XB index. The objective is therefore to minimise the XB index for achieving proper clustering. The ideal value for the XB index is 0.

***"DB"* Index:** Davies-Bouldin index [56] is a function of the ratio of the sum of within-cluster scatter to between-cluster separation. The scatter within i-th cluster, S_i, is computed as $S_i = \frac{1}{|X_i|}\sum_{x \in X_i} \|x - z_i\|$ and the distance between cluster X_i and X_j denoted by d_{ij}, is defined as $d_{ij} = \|z_i - z_j\|$. Here, z_i represents the i-th cluster centre. The DB index is defined as:

$$DB = \frac{1}{K}\sum_{i=1}^{K} R_i \tag{A.0.7}$$

where $R_i = max_{i \neq j} R_{ij}$, $R_{ij} = (\frac{S_i + S_j}{d_{ij}})$. The objective is to minimise the DB index for achieving proper clustering. The ideal value for the DB index is 0.

Appendix B

Stability indices

Stability indices are used to measure the stability of clustering results. The following are five stability indices which used in chapter 6. Four of these are derived from statistics and similarity measure of partitions. There are several similarity measures for partitions of a finite set [2, 65]. For two partitions P_1 and P_2, a pair of data points (x_i, x_j) from the dataset is referred using the following terms:

- **SS**: if both points belong to the same cluster of P_1 and P_2.
- **SD**: if points belong to the same cluster of P_1 and to different cluster of P_2.
- **DS**: if points belong to different clusters of P_1 and to the same cluster of P_2.
- **DD**: if both points belong to different clusters of P_1 and to different clusters of P_2.

Let a, b, c and d are the number of SS, SD, DS and DD pairs respectively, then $a+b+c+d = M$ which is the maximum number of pairs in the dataset, i.e., $M = n(n-1)/2$, where n is the total number of points in the dataset. Now the indices to measure the degree of similarity between P_1 and P_2 can be defined as:

$$Rand\,Statistic = \frac{a+b}{M} \tag{B.0.1}$$

$$Jaccard\,Coefficient = \frac{a}{a+b+c} \tag{B.0.2}$$

$$Folkes\,and\,Mallows = \sqrt{\frac{a}{a+b} \times \frac{a}{a+c}} = \frac{a}{\sqrt{m_1 m_2}} \tag{B.0.3}$$

where $m_1 = (a+b)$, $m_2 = (a+c)$.

$$Hubert\,Statistic = \frac{M\,a - m_1 m_2}{\sqrt{m_1 m_2 (M - m_1)(M - m_2)}} \tag{B.0.4}$$

In case of *Rand* index, the overall similarity of r partitions is defined as follows:

$$Overall\ Rand\ similarity\ index = \frac{1}{r(r-1)} \sum_{i=1}^{r} \sum_{j=i+1}^{r} Rand(P_i, P_j) \tag{B.0.5}$$

Similarly, Eq. B.0.5 can be applied on other similarity indices, i.e., one can obtain four stability indices. It should be noted that higher values of the overall similarity indices indicate better clustering stability. The ideal normalised value of each of these four stability indices is 1.

Bootstrapping method

Law *et al.* [67] uses bootstrapping to estimate the variability of a clustering algorithm by considering it as an estimator for the partition of the data space. If a partition is valid, its variability should be low. This is achieved by generating r bootstrap samples, each of size s, by sampling the data, then apply the clustering algorithm on each sample to obtain the corresponding partition. Then, the variability of the i-th clustering algorithm can be defined as:

$$B = \frac{1}{r-1} \sum_{j=1}^{r} d(P_j, A^*) \tag{B.0.6}$$

where B is a measure of variability, P_j is the partition obtained by running the i-th clustering algorithm on bootstrap sample j, A^* is the average partition, and d is the distance measure. The lower values of the B index indicate better clustering stability. The ideal value of the B index is 0.

Bibliography

[1] A. Webb, *Statistical Pattern Recognition.* Chichester: John Willey & Sons, 2003.

[2] A. Jain and R. Dubes, *Algorithms for Clustering Data.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[3] S. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, pp. 241–254, September 1967.

[4] C. Wallace and D. Boulton, "An information measure for classification," *Computer Journal*, vol. 11, pp. 185–194, August 1968.

[5] B. Everitt, S. Landau, and M. Leese, *Cluster Analysis.* London: Arnold, 2001.

[6] L. Kaufman and P. Rousseeuw, *Finding Groups in data: an introduction to cluster analysis.* New York: John Wiley & Sons, 1990.

[7] S. Theodoridis and K. Koutroubas, *Pattern Recognition.* San Diego: Academic Press, 2003.

[8] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, pp. 645–677, May 2005.

[9] A. Jain, M. Murty, and P. Flyn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, pp. 264–323, September 1999.

[10] A. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 4–37, January 2000.

[11] K. Cios, W. Pedrycz, and R. Swiniarski, *Data Mining Methods for Knowledge Discovery*. Boston: Kluwer Academic, 1998.

[12] P. Mahalanobis, "On the generalized distance in statistics," *National Academy of Sciences*, vol. 12, pp. 49–55, 1936.

[13] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[14] A. Fielding, *Cluster and Classification Techniques for the Biosciences*. Cambridge: Cambridge University Press, 2007.

[15] T. Zhang, R. Ramakrishnan, and M. Linvy, "BIRCH: A new data clustering algorithm and its applications," *Data Mining and Knowledge Discovery*, vol. 1, pp. 141–182, June 1997.

[16] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," *ACM SIGMOD international conference on Management of the data*, pp. 73–84, 1998.

[17] G. Karypis, E. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, pp. 68–75, August 1999.

[18] T. Ng and H. Jiawei, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, pp. 1003–1016, September/October 2002.

[19] M. Yang, "A survey of fuzzy clustering," *Mathematical and Computer Modelling*, vol. 18, no. 11, pp. 1–16, 1993.

[20] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition - part I," *IEEE Transactions on Systems, Man, and Cybernetics-PART B*, vol. 29, pp. 778–785, December 1999.

[21] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence, vol. 24, pp. 881–892, July 2002.

[22] S. Mu-Chun and C. Chien-Hsing, "A modified version of the k-means algorithm with a distance based on cluster symmetry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 674–680, June 2001.

[23] K. Wagstaff, S. Rogers, and S. Schroedl, "Constrained k-means clustering with background knowledge," in *Proceedings of 18th International Conference on Machine Learning (ICML-01)*, (Williams College, Williamstown, MA, USA), pp. 577–584, 28 June - 1 July 2001.

[24] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, (Portland, OR, USA), pp. 226–231, 2-4 August 1996.

[25] A. Hinneburg and D. Keim, "An efficient approach to clustering in large multimedia databases with noise," in *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-98)*, (New York, USA), pp. 58–65, August 1998.

[26] W. Wang, J. Yang, and R. Muntz, "STING: A statistical information grid approach to spatial data mining," in *the 23rd International Conference on Very Large Data Bases (VLDB-97)*, (Athens, Greece), pp. 186–195, 25-29 August 1997.

[27] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: A multi-resolution clustering approach for very large spatial databases," in *the 24th International Conference on Very Large Data Bases (VLDB-98)*, (New York, USA), pp. 428–439, 24-27 August 1998.

[28] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. New York: Wiley, 1997.

[29] T. Kohonen, *Self-Organizing Maps*. New York: Springer-Verlag, 1997.

[30] U. V. Luxburg, "A tutorial on spectral clustering," Tech. Rep. TR-149, In Max Planck Institute for Biological Cybernetics, 2006.

[31] B. Fischer and J. Buhmann, "Path based clustering for grouping smooth curves and texture segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1–6, April 2003.

[32] H. Zhexue, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 2, pp. 283–304, September 1998.

[33] G. Sudipto, R. Rajeev, and S. Kyuseok, "ROCK: A robust clustering algorithm for categorical atributes," in *Proceedings of the 15th International Confererence on Data Engineering (ICDE-99)*, (Sydney, NSW, Australia), pp. 512–521, 23-26 March 1999.

[34] A. Bouchachia and W. Pedrycz, "Data clustering with partial supervision," *Data Mining and Knowledge Discovery*, vol. 12, pp. 47–78, January 2006.

[35] W. Pedrycz and J. Waletzky, "Fuzzy clustering with partial supervision," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 27, pp. 787–795, October 1997.

[36] W. Pedryez, "Algorithms of fuzzy clustering with partial supervision," *Pattern Recognition Letters*, vol. 3, pp. 13–20, January 1985.

[37] K. Doi, "Current status and future potential of computer-aided diagnosis in medical imaging," *British Journal of Radiology*, vol. 78, pp. 3–19, 2005.

[38] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proceedings of the 19th International Conference on Machine Learning*, (Sydney, Australia), pp. 19–26, July 2002.

[39] A. Blum, J. Lafferty, M. Rwebangira, and R. Reddy, "Semi-supervised learning using randomized mincuts," in *Proceedings of the 21th International Conference on Machine Learning*, (Banff, Alberta, Canada), pp. 92–100, 4-8 July 2004.

[40] X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty, "Nonparametric transforms of graph kernels for semi-supervised learning," *Advances in Neural Information Processing Systems*, vol. 17, pp. 1641–1648, December 2005.

[41] A. Demiriz, K. Bennett, and M. Embrechts, "Semi-supervised clustering using genetic algorithms," *Intelligent Engineering Systems*, vol. 9, pp. 809–814, 1999.

[42] B. Jeon and D. Landgrebe, "Partially supervised classification using weighted unsupervised clustering.," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, pp. 1073–1079, March 1999.

[43] S. Basu, M. Bilenko, and R. Mooney, "A probablistic framework for semi-supervised clustering," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004)*, (Seatle, WA), pp. 59–68, August 2004.

[44] K. Nigam, A. McCallum, S. Thrunand, and T. Mitchell, "Text classification from labeled and unlabeled documents using expectation-maximization," *Machine Learning*, vol. 39, pp. 103–134, May 2000.

[45] L. Hunter and D. States, "Bayesian classification of protein structure," *IEEE Expert: Intelligent Systems and Their Applications*, vol. 7, pp. 67–75, August 1992.

[46] C. Pizzuti and D. Talia, "P-autoclass: scalable parallel clustering for mining large data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, pp. 629–641, May 2003.

[47] D. Foti, D. Lipari, C. Pizzuti, and D. Talia, "Scalable parallel clustering for data mining on multicomputers," in *the 3rd Workshop on High Performance Data Mining in conjunction with International Parallel and Distributed Processing Symposium 2000 (IPDPS'00)*, (Cancun, Mexico), pp. 390–398, May 2000.

[48] I. Dhillon and D. Modha, "A data-clustering algorithm on distributed memory multiprocessors," in *Large-Scale Parallel Data Mining, Lecture Notes in Artificial Intelligence*, pp. 245–260, March 2000.

[49] D. Judd, P. McKinley, and A. Jain, "Large-scale parallel data clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 871–876, August 1998.

[50] C. Olson, "Parallel algorithms for hierarchical clustering," *Parallel Computing*, vol. 21, no. 8, pp. 1313–1325, 1993.

[51] A. Petrosino and M. Verde, "P-AFLC: a parallel scalable fuzzy clustering algorithm," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR-2004)*, (Cambridge, UK), pp. 809–812, 23-26 August 2004.

[52] A. Garg, A. Mangla, N. Gupta, and V. Bhatnagar, "PBIRCH: A scalable parallel clustering algorithm for incremental data," in *the 10th International Database Engineering and Applications Symposium (IDEAS-06)*, (Antibes, France), pp. 315–316, 18-22 September 2006.

[53] S. Brecheisen, H. Kriegel, and M. Pfeifle, "Parallel density-based clustering of complex objects," in *the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-06)*, vol. 3918, (Singapore), pp. 179–188, 9-12 April 2006.

[54] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part I," in *SIGMOD Record*, vol. 31, June 2002.

[55] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part II," in *SIGMOD Record*, vol. 31, September 2002.

[56] D. Davies and D. Bouldin, "A cluster separation measure," *IEEE Tranactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.

[57] J. Dunn, "A fuzzy relative of isodata process and its use in detecting compact well-separated clusters," *Journal of Cybernetics*, vol. 3, pp. 32–57, 1973.

[58] R. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, pp. 1–27, 1974.

[59] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1650–1654, December 2002.

[60] G. Milligan and C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, pp. 159–179, June 1985.

[61] C. Fraley and A. Raftery, "How many clusters? which clustering method? answers via model-based cluster analysis," *The Computer Journal*, vol. 41, no. 8, pp. 578–588, 1998.

[62] J. Bezdek and N. Pal, "Some new indexes of cluster validity," *IEEE Transactions on Systems, Man, and Cybernetics-PART B*, vol. 28, pp. 301–315, June 1998.

[63] L. Hall, I. Ozyurt, and J. Bezdek, "Clustering with a genetically optimized approach," *IEEE Transactions on Evoluationary Computation*, vol. 3, pp. 103–112, July 1999.

[64] X. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 841–847, August 1991.

[65] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *Journal of Intelligent Information Systems*, vol. 17, pp. 107–145, December 2001.

[66] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the Gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[67] M. Law and A. Jain, "Cluster validity by bootstrapping partitions," Tech. Rep. MSU-CSE-03-5, Department of Computer Science and Engineering Michigan State University, 2003.

[68] T. Lange, M. Braun, and J. Buhmann, "Stability-based validation of clustering solutions," *Neural Computation*, vol. 16, pp. 1299–1323, June 2004.

[69] A. Ben-Hur, A. Elisseeff, and I. Guyon, "A stability based method for discovering structure in clustered data," in *Pacific Symposium on Biocomputing*, vol. 7, (Lihue, Hawaii, USA), pp. 6–17, January 2002.

[70] E. Levine and E. Domany, "Resampling method for unsupervised estimation of cluster validity," *Neural Computation*, vol. 13, pp. 2573–2593, November 2001.

[71] A. Jain and J. Moreau, "Bootstrap techniques in cluster analysis," *Pattern Recognition*, vol. 20, pp. 547–568, 1987.

[72] R. Tibshirani, G. Walther, D. Botstein, and P. Brown, "Cluster validation by predication strength," Tech. Rep. CA 94305, Statistics Department, Stanford University, 2001.

[73] S. Dudoit and J. Fridlyand, "A predication-based resampling method for estimating the number of clusters in a data set," *Gnome Biology*, vol. 3, no. 7, 2002. Available on-line: `http://genomebiology.com/2002/3/7/research/0036`.

[74] T. Lange, M. Braun, V. Roth, and J. Buhmann, "Stability-based model selection," *Advances in Neural Information Processing Systems*, vol. 15, pp. 617–624, 2003.

[75] J. Proakis, *Digital Communications*. Boston, USA: McCraw-Hill, 2001.

[76] "UCI repository of machine learning databases." `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[77] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, (Cambridge, MA), pp. 849–856, MIT Press, 2002.

[78] T. Cormen, C. Leiserson, L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press and McCraw-Hill, 2001.

[79] J. Fonseca and M. Cardoso, "Mixture-model cluster analysis using information theoretical criteria," *Intelligent Data Analysis*, vol. 11, no. 2, pp. 155–173, 2007.

[80] B. Kverh and A. Leonardis, "A generalisation of model selection criteria," *Pattern Analysis and Applications*, vol. 7, pp. 51–65, April 2004.

[81] X. Hu and L. Xu, "Investigation on several model selection criteria for determining the number of clusters," *Neural Information Processing - Letters and Reviews*, vol. 4, pp. 1–10, July 2004.

[82] "Leukemia dataset." `http://sdmc.lit.org.sg/GEDatasets/Datasets.html`.

[83] "The STARE project." `http://www.ces.clemson.edu/~ahoover/stare`.

[84] N. Salem and A. Nandi, "Segmentation of retinal blood vessels using scale space features and k-nearest neighbour classifier," in *the 31st International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2006)*, (Toulouse, France), 14-19 May 2006.

[85] T. Liao, C. Ting, and P. Chang, "An adaptive genetic clustering method for exploratory mining feature vector and time series data," *International Journal of Production Research*, vol. 44, pp. 2731–2748, July 2006.

[86] S. Salem and A. Nandi, "New assessment criteria for clustering algorithms," in *IEEE international workshop in Machine Learning For Signal Processing (MLSP 2005)*, (Mystic, CT, USA), pp. 285–290, 28-30 September 2005.

[87] T. Fawcett, "ROC graphs: notes and practical considerations for researchers," Tech. Rep. HPL-2003-4, HP Laboratories, 2004.

[88] A. Hoover, V. Kouznetsova, and M. Goldbaum, "Locating blood vessels in retinal images by piecewise thresholding probing of a matched filter response," *IEEE Transactions on Medical Imaging*, vol. 19, pp. 203–210, March 2000.

[89] H. Guo and A. Nandi, "Breast cancer diagnosis using genetic programming generated feature," *Pattern Recognition*, vol. 39, pp. 980–987, May 2006.

[90] R. Duha, P. Hart, and D. Stork, *Pattern Classification*. Chichester: John Willey & Sons, Inc., 2001.

[91] C. Hsu and C. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, pp. 415–425, March 2002.

[92] J. Kishore, L. Patnaik, V. Mani, and V. Arawal, "Application of genetic programming for multicategory pattern classification," *IEEE Transactions on Evoluationary Computation*, vol. 4, pp. 242–258, September 2002.

[93] W. Gropp, S. Huss-Lederman, A. Lumsdaine, E. Lusk, B. Nitzberg, W. Saphir, and M. Snir, *MPI: The Complete Reference*. Cambridge, MA: The MIT Press, 1998.

[94] M. Niemeijer, J. Staal, B. van Ginneken, M. Long, and M. Abramoff, "Comparative study of retinal vessel segmentation methods on a new publicly available database," in *SPIE Medical Imaging*, vol. 5370, pp. 648–656, May 2004.

[95] S. Salem, N. Salem, and A. Nandi, "Segmentation of retinal blood vessels using a novel clustering algorithm (RACAL) with a partial supervision strategy," *Medical and Biological Engineering and Computing*, vol. 45, pp. 261–273, March 2007.

[96] M. Eisen, B. Spellman, P. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," in *National Academy of Sciences*, vol. 95, (USA), pp. 14863–14868, December 1998.

[97] R. Herwig, A. Poustka, C. Müller, C. Bull, H. Lehrach, and J. O'Brien, "Large-scale clustering of cDNA-fingerprinting data," *Genome Research*, vol. 9, pp. 1093–1105, November 1999.

[98] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dimitrovsky, S. Lander, and T. Golub, "Interpreting patterns of gene expression with self-organizing maps:

Methods and application to hematopoietic differentiation," *National Academy of Sciences*, vol. 96, pp. 2907–2912, March 1999.

[99] D. Dembélé and P. Kastner, "Fuzzy c-means method for clustering microarray data," *Bioinformatics*, vol. 19, pp. 973–980, May 2003.

[100] M. Rhouma and H. Frigui, "Self-organization of pulse-coupled oscillators with application to clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 1–16, February 2001.

[101] S. Salem and A. Nandi, "Novel clustering algorithm (RACAL) and a partial supervision strategy for classification," in *IEEE international workshop in Machine Learning For Signal Processing (MLSP 2006)*, (Mynooth, Ireland), pp. 313–318, 06-08 September 2006.

[102] B. Lam and H. Yan, "Assessment of microarray data clustering results based on a new geometrical index for cluster validity," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 11, pp. 341–348, February 2007.

[103] K. Yeung, D. Haynor, and W. Ruzzo, "Validating clustering for gene expression data," *Bioinformatics*, vol. 17, no. 4, pp. 309–318, 2001.

[104] N. Bolshakova and F. Azuaje, "Cluster validation techniques for genome expression data," *Signal Processing*, vol. 83, pp. 835–833, April 2003.

[105] N. Speer, C. Spieth, and A. Zell, "Biological cluster validity indices based on the gene ontology," in *Lecture Notes in Computer Science*, vol. 3646, pp. 429–439, Springer, 2005.

[106] F. Azuaje, "A cluster validity framework for genome expression data," *Bioinformatics*, vol. 18, no. 2, pp. 319–320, 2002.

[107] "Stanford yeast cell cycle analysis project." `http://genome-www.stanford.edu/cellcycle/`.

[108] P. T. Spellman, G. Sherlock, M. Quang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive identification of cell-cycle regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization," *Molecular Biology of the Cell*, vol. 9, pp. 3273–3297, December 1998.

[109] R. Cho, M. Campbell, E. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. Wolfsberg, A. Gabrielian, D. Landsman, D. Lockhart, and R. Davis, "A genome-wide transcriptional analysis of the mitotic cell cycle," *Molecular Cell*, vol. 2, pp. 65–73, July 1998.

[110] A. Alizadeh, M. Eisen, R. Davis, C. Ma, I. Losses, A. Rosenwald, J. Boldrick, H. Sabet, T. Tran, X. Yu, J. Powell, L.Yang, G. Marti, T. Moore, J. Hudson, L. Lu, D. Lewis, G. S. R. Tibshirani, W. Chan, T. Greiner, D. Weisenburger, J. Armitage, R. Warnke, R. Levy, W. Wilson, M. Grever, J. Bird, D. Botstein, P. Brown, and M. Staudt, "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, pp. 503–511, Feburary 2000.

[111] "Lymhoma dataset." `http://llmpp.nci.nih.gov/lymphoma/`.

[112] M. Eisen and P. Brown, "DNA arrays for analysis of gene expression," *Methods in Enzymology*, vol. 303, pp. 179–205, 1999.

[113] X. Chen, S. Cheung, S. So, S. Fan, C. Barry, J. Higgins, K. Lai, J. Ji, S. Dudoit, I. Ng, M. Rijn, D. Botstein, and P. Brown, "Gene expression patterns in human liver cancers," *Molecular Biology of the Cell*, vol. 13, pp. 1929–1939, June 2002.

[114] "Liver cancer dataset." `http://genome-www.stanford.edu/hcc/`.

[115] G. Sherlock, T. Hernandez-Boussard, A. Kasarskis, G. Binkley, J. Matese, S. Dwight, M. Kaloper, S. Weng, H. Jin, C. Ball, M. Eisen, P. Spellman, P. Brown, D. Botstein, and J. Cherry, "The stanford microarray database," *Nucleic Acids Research*, vol. 29, no. 1, pp. 152–155, 2001.

[116] C.-H. Chou, M.-C. Su, and E. Lai, "A new cluster validity measure and its application to image compression," *Pattern Analysis and Applications*, vol. 7, pp. 205–220, July 2004.

[117] P. Rousseeuw, "Silhouette: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, November 1987.

[118] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gassenbeck, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531–537, October 1999.

[119] X. Wen, S. Fuhrman, G. Michaels, D. Carr, S. Smith, J. Barker, and R. Somogyi, "Large-scale temporal gene expression mapping of central nervous system development," *National Academy of Sciences*, vol. 95, pp. 334–339, January 1998.

[120] N. Belacel, Q. Wang, and M. Cuperlovic-cluf, "Clustering methods for microarray gene expression data," *Journal of Integrative Biology*, vol. 10, no. 4, pp. 507–531, 2006.

[121] L. Li, C. Weinberg, T. Darden, and L. Pedersen, "Gene selection for sample classification based on gene expression data: Study of sensitivity to choice of parameters of GA/KNN method," *Bioinformatics*, vol. 17, no. 12, pp. 1131–1142, 2001.

[122] J. Hong and S. Cho, "Lymphoma cancer classification using genetic programming with SNR features," in *Lecture Notes in Computer Science*, vol. 3003, pp. 78–88, Springer, April 2004.

[123] D. Nguyen and D. Rocke, "Tumor classification by partial least squares using microarray gene expression data," *Bioinformatics*, vol. 18, pp. 39–50, January 2002.