2014

An Unsupervised Consensus Control Chart Pattern Recognition Framework

Siavash Haghtalab
University of Central Florida

AN UNSUPERVISED CONSENSUS CONTROL CHART PATTERN RECOGNITION FRAMEWORK

by

SIAVASH HAGHTALAB
M.Sc. University of Central Florida
Orlando, Florida, 2014
B.Sc. Tehran Polytechnic
Tehran, Iran, 2012

A thesis submitted in partial fulfilment of the requirements
for the degree of Master of Science
in the Department of Industrial Engineering and Management Systems
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2014

ABSTRACT

Early identification and detection of abnormal time series patterns is vital for a number of manufacturing. Slide shifts and alterations of time series patterns might be indicative of some anomaly in the production process, such as machinery malfunction. Usually due to the continuous flow of data monitoring of manufacturing processes requires automated Control Chart Pattern Recognition (CCPR) algorithms. The majority of CCPR literature consists of supervised classification algorithms. Less studies consider unsupervised versions of the problem. Despite the profound advantage of unsupervised methodology for less manual data labeling their use is limited due to the fact that their performance is not robust enough for practical purposes. In this study we propose the use of a consensus clustering framework. Computational results show robust behavior compared to individual clustering algorithms.

ACKNOWLEDGMENTS

I would like to express my special appreciation and thanks to my advisor Dr. Petros Xanthopoulos, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and guiding me throughout my academic career. I would also like to thank my committee members, Dr. Jennifer Pazour, Dr. Luis Rabelo and Dr. Kaveh Madani for their guidances and serving as my committee members even at hardship. I also want to thank you for letting my defense be an enjoyable moment, and for your comments and suggestions, thanks to you.

Last but not the least, I would like to thank my family for supporting me spiritually throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

In today's world different techniques of Data Mining are used widely for different types of problems. Data Mining is an interdisciplinary subfield of computer science which discovers patterns and extracts information from large datasets.

Data Mining is strongly connected with statistics and science of numbers. Many techniques used in Data Mining are statistical techniques.

Machine learning is a very related field with data mining which is a branch of artificial intelligence and deals with construction of machines that can learn from data. In 1959 Arthur Samuel defined machine learning as "Field of study that gives computers the ability to learn without being explicitly programmed".

There are several techniques of Machine Learning used in problems and they can be divided into two main categories: Supervised Learning and Unsupervised Learning.

In supervised learning there is a prior information about data which are presented as labels. In supervised learning we have two sets of Train Set and Test Set. In train set each pair contains the input and desired output. With the help of train set, algorithm will construct the function and can predict or label new data. There are several supervised methods such as Artificial Neural Networks, Support Vector Machines and Regression Analysis.An example of supervised learning is prediction of house prices for the next 5 year. By getting data from the previous years prices we can find a function that fits best the data with the least error and make predictions for future years with a certain amount of error.

On the other side, unsupervised learning is the task of classification without any prior information

or labeled data. The task in unsupervised learning is to find the hidden structurer in unlabeled data. Unsupervised learning and Clustering are the terms used interchangeably. In clustering we are trying to find the hidden structure in data and group the data based on their similarities. Because there is no label or prior information of the data, there is no reward or penalty involved, clustering is used only when there is no prior information about the data. An example of clustering is in marketing, in this case, we face a large data set of customers and we are interested to find their common behavior or interest for marketing strategies.

Different clustering algorithms such as K-Means, Fuzzy clustering, Spectral Clustering and Hierarchical Clustering are proposed which we will go over them in details in chapter 2.

Clustering has big usages in image segmentation, biology, business and marketing, socio economics and so many other fields when there is a need to group and make use of unlabeled data.

In this study we are going to purpose a new method of Unsupervised Learning (Clustering) for unlabeled data.

First we will give a description of clustering and some of its applications. Then for our method, we will describe a clustering method called *Consensus Clustering* and apply that to our application. At the end, based on the evaluation methods used in literature, we will evaluate the quality of our proposed method and give the results and conclusions.

In the rest of the chapter, first we will talk about the task of clustering in details, next we will go through consensus clustering and some ensemble methods and then we will study Control Chart Pattern Recognitions (CCPR).

In chapter 2 we will talk about our proposed method which is an unsupervised framework for classifying binary class data. In the same chapter we will also talk about some evaluation methods for our proposed method to measure the quality of the clusterings.

Finally, in chapter 3 and 4 we will give the findings, performance of the proposed method and conclusions.

Clustering

Clustering is the task of grouping data into groups so that similar data are in the same group. The objective is to maximize the within the group similarity and minimize the between the group similarity.

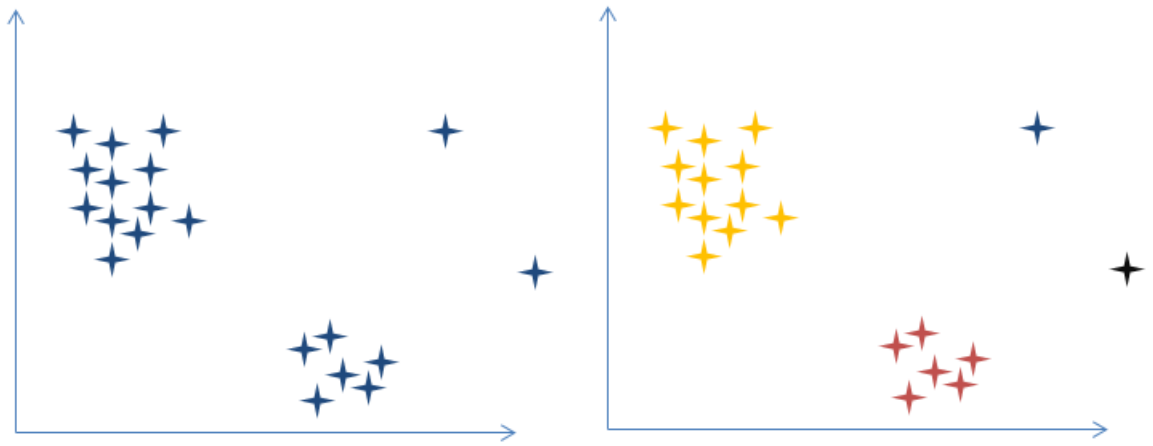
Clustering is very useful in data mining, document retrieval, image segmentation and pattern classification. In the literature there are lots of attention paid to clustering and its applications.

Figure1.1 is an example of clustering, in figure1.1.a original data points are shown and figure1.1.b is an example of a clustering of these points. As shown in the figure, the effort is to group similar and closer data together. There are different methods of clustering based on the data type and each will give a different result.

Only when there is a little prior information about the data or the decision maker must take as few assumption about the data as possible we use clustering.

A full review of clustering is given by Jain et al. (1999). In their paper, they have defined five different steps for clustering as follows:

1. Pattern representation
2. Definition of a pattern proximity measure appropriate to the data domain
3. Clustering



(a) (b)

Figure 1.1: An example of clustering

4. Data abstraction (if needed)

5. Assessment of output (if needed)

The first step is to identify the number of clusters or groups and scale of features available for clustering algorithm. Usually, data are consist of different number of features, for example a manufacturing data might contain features about dimensions, weights and materials used. Number of features or number of clusters might not be identifiable or controllable by the user but there are methods to fit the best number of clusters. Liu et al. (2010) did an study on internal clustering measures, there are different methods studied and they can be used together to find the best number of clusters, but we will not go through them in this study since our problem is a binary classification problem.

In the second step we define similarity measures for the clustering, there are different similarity measures existing for different types of problems such as euclidean distance. It is very important

to use the proper similarity measure since not using a good fit will result in bad clusterings.

In the clustering step we can use different clustering algorithms for grouping different types of data. In general there are two types of hard clusterings and soft clusterings. Hard clusterings group data into defined groups and data can belong to only one specific group, in soft clusterings each data point belongs to a group with a degree of membership and might belong to some other group with another degree of membership.

In data abstraction step we are trying to find simple representation of the data.

In the last step, we are interested to validate and measure the goodness of clusterings. First it is important to test the data itself and see if it is clusterable. This task is called *Cluster Tendency*. Dubes (1987) did a Monte Carlo study on the relative effectiveness of two internal indices in estimating the true number of clusters. There are methods proposed in literature which we will not go through here. *Cluster Validation* is also when we assess the quality or goodness of clustering. Clustering Validation measures are subjective to the matter, based on different problems and data there are different clustering validation measures. In chapters 3 and 4 we will talk about some of these methods that we used for our proposed algorithm.

Clustering algorithms can be divided into four types:

1. Hierarchical Clusterings
2. Centroid-Based Clusterings
3. Distribution-Based Clusterings
4. Density-Based Clusterings

Hierarchical based clusterings build the clusters based on a hierarchy. First each data will be in its own cluster and in each step based on a distance measure, clusters will be merged until the algorithm reaches a single cluster. Based on the desired number of clusters, the algorithm can be stopped or cut at any step.

In centroid-based clusterings such as k-means, first some initial centers will be defined and the data will be clustered based on their distances from the centers (a function similar to Voronoi Diagram). These algorithms are usually iterative, in each iteration the centers are updated until the algorithm reaches the optimal solution where the distances are minimized. These algorithms might not give the optimal solution.

Distribution models are most close to statistical models, they use statistical distributions for clustering.

Density-based clustering work with assigning points to clusters as dense areas and the rest of the points will be considered as noise.

Clustering is necessary when data is unlabeled regardless of whether the data is binary, categorical, numerical, interval, ordinal, relational, textual, spatial, temporal, spatio-temporal, image, multimedia, or mixtures of the above data types (Warren Liao, 2005). There are different choices of clustering algorithms depending on the type of data and application as mentioned above.

Data are called static if all their feature values do not change negligibly over time. So many clustering analyses has been performed on these types of data. Aside from the above given classification, Han et al. (2006) classified clustering methods for handling static data into five major categories:

1. Partitioning Methods
2. Hierarchical Methods

3. density-Based Methods

4. Grid-Based Methods

5. Model-Based Methods

Each of these types of algorithms were used for different applications. Gustafson and Kessel (1978) proposed a fuzzy clustering algorithm using a fuzzy covariance matrix. Fuzzy sets were used in clustering first by Bellman et al. (1966), fuzzy clustering is also used for several medical diagnosis (Adey, 1972) and in cardiovascular research (Kalmanson and Stegall, 1975).

Another method of clustering is spectral clustering which uses the concept of spectrum and eigenvectors. Verma and Meila (2003) gave a comparison between spectral clustering algorithms and some of their applications. Shi and Malik proposed a spectral clustering algorithm for image segmentation (Shi and Malik, 2000), Jordan and Weiss also proposed another algorithm (Y et al., 2001) which we will use later on.

Another method of clustering is Hierarchical clustering, so many works related to biological sciences are done on this matter, Johnson (1967) have developed a hierarchical clustering algorithm with a distance measure. Bandyopadhyay and Coyle (2003) also presented a hierarchical clustering algorithm for wireless sensor networks.

In another article, a scalable method for clustering of time series data is presented which uses some global measures instead of the points themselves (Wang et al., 2004). Lin et al. (2004) presented an iterative incremental partitioning method for clustering of time series data.

Xiong and Yeung proposed a model based approach for time series with different lengths using autoregressive moving average (ARMA) (Xiong and Yeung, 2004). Characteristic based clustering is also proposed by Wang et al. (2006). in another article a hybrid time series clustering algorithm

that uses dynamic time warping and hidden Markov model is presented (Oates et al., 1999).

Consensus Clustering

Different clusterings of same data can be obtained from different clustering algorithms or different runs of a nondeterministic clustering algorithm. Consensus clustering combines different clusterings into one consensus that is the representative of those clusterings, and this emphasizes the common organization.

For instance, k-means clustering gives different results for the same data by choosing different initial centers and by use of consensus clustering, here we can combine the results of some weak clusterings into a good clustering which will also be a representative of them.

Consensus clustering has a very useful roll because of the prevalence of large data sets and their availability from different sources. In general, in consensus clustering we are trying to use variability in clustering and data, this variability might result form the clustering algorithms, the data itself or local optimality.

Numbers of works are done on this matter, Goder and Filkov (2008) have implemented a number of heuristics for the consensus clustering problem and compared their performance, independent of data size, in terms of efficacy and efficiency, on both simulated and real data sets. Li et al. (2004) shows different clustering criteria and their connection with consensus. Lancichinetti and Fortunato (2012) showed that consensus clustering can be combined with any existing method in a self-consistent way, enhancing considerably both the stability and the accuracy of the resulting partitions. In another paper Li et al. (2007) showed how consensus and semi-supervised clustering can be formulated within the framework of nonnegative matrix factorization. Weighted consensus

clustering is proposed by Li and Ding (2008), in their approach, different weights are given to different clustering inputs. A complete survey of ensemble algorithms is given in(Vega-Pons and Ruiz-Shulcloper, 2011) and different methods are compared in (Ghaemi et al., 2009)

Consensus clustering can be divided into three steps as shown in figure1.2:

1. Data Gathering
2. Clustering
3. Consensus Function

In the first step we will gather the required data for our problem. Here we might have data from different categories of individuals, different sources, different measurements or data with variety in general. In the second step we run the clusterings on the data, in this step we might choose to run several clustering algorithms or run the same algorithm for several times. At the last step by defining a consensus function we will decide how to use the clustering in order to construct the consensus clustering.

Choosing the different algorithms and data source depends on the specific problem, but later we will show that even by using bad algorithm, consensus can still give good results.

In general, if there is no variety in clustering or data, then there is no point in using a consensus method since they are usually more time consuming and they need more resources.

Different approaches are studied for the consensus function which combines the different clusterings. Some functions use the *Pick-A-Cluster* method which simply picks a random algorithm

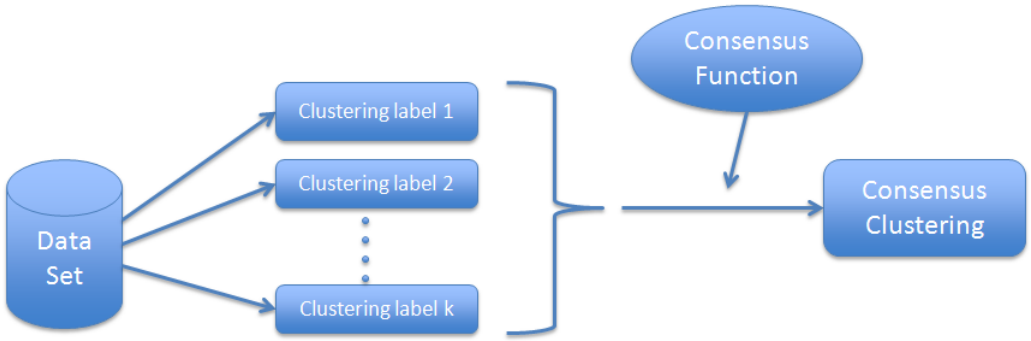


Figure 1.2: Consensus Clustering Scheme

each time. There is also the *Average Linkage* method, in this method each data belongs to its own cluster and in each step two closest clusters merge together and form a single cluster until the the average sdd between clusters is 1/2. A very common approach which we will use in here is voting. In voting based on the number of clusterings, points are clustered to their highest vote.

In (Kuncheva et al., 2006) 24 methods for creating the consensus is studied and those methods are evaluated on 24 data.

As shown in figure1.2, in voting, different clusterings can be obtained form different runs of a same algorithm or from different clustering algorithms. Consensus function will combine and process these clustering labels and create the consensus graph. In the most of the cases, several runs of a single algorithm or a single run on several data is studied. Here we are going to combine different clustering algorithms on a same data and construct the consensus based on the labels we get from those clusterings.

As an example, if we have four different clustering of a data:

$$\begin{aligned}\gamma_1 &= \{1,1,0,1,0,1,1,0,0\}\\ \gamma_2 &= \{0,1,1,1,1,1,0,0,0\}\\ \gamma_3 &= \{1,1,1,1,1,0,0,0,0\}\\ \gamma_4 &= \{0,1,1,0,1,1,0,0,1\}\end{aligned} \tag{1.1}$$

Then based on voting, consensus clustering would be:

$$\gamma_* = \{1,1,1,1,1,1,0,0,0\} \tag{1.2}$$

As we can see, not all the clusterings are giving the same result, some of them might have errors but by using consensus we were able to get a final clustering which has a common structure with the others and has better quality.

Consensus clustering can also be formulated as an optimization problem(Grötschel and Wakabayashi, 1989).

For n different clustering labels P_k for $k = 1, 2, ..., n$ we define $r_{ij}^{(k)} = 1$ if samples s_i and s_j belong to the same cluster in clustering P_k and 0 if they don't. Then the decision variable r_{ij} is defined the same, the objective function is as follows:

$$\sum_{i=1}^{k} d(P, P_i) = \sum_{i=1}^{k} \sum_{i,j} (r_{ij}^{(k)} - r_{ij})^2 \tag{1.3}$$

d is the distance function between two clusterings. Since $r_{ij}^{(k)}$ and $r_{ij} \in 0, 1$ then the function will be linearized:

$$\sum_{k}\sum_{i,j} r_{ij}^{(k)} + \sum_{k}\sum_{i,j}(1 - 2.r_{ij}^{(k)})r_{ij} \tag{1.4}$$

It can be observed that the objective is 0-1 linear and can be written as:

$$c + \sum_{i,j} c_{ij} r_{ij} \tag{1.5}$$

where:

$$c = \sum_{k}\sum_{i,j} r_{ij}^{(k)}, c_{ij} = \sum_{k}(1 - 2.r_{ij}^{(k)}) \tag{1.6}$$

At the end the optimization function can be formulated as:

$$min \sum_{ij} C_{ij} r_{ij} \tag{1.7a}$$

$$s.t.\ r_{ii} = 1, \qquad i = 1, ..., n \tag{1.7b}$$

$$r_{ij} = r_{ji}, \qquad i, j = 1, ..., n \tag{1.7c}$$

$$r_{ij} + r_{jk} - r_{ik} \leq 1, \qquad i, j, k = 1, ..., n \tag{1.7d}$$

$$r_{ij} \in \{0, 1\}, \qquad i, j = 1, ..., n \tag{1.7e}$$

Since $r_{ij} = r_{ji}$ these variables can be replaced by x_{ij} and r_{ii} can be dropped since it is a fix variable.We define weights $w_{ij} = c_{ij} + c_{ji}$. The problem can be formulated as below:

$$min \sum_{1 \leq i < j \leq n} w_{ij}.x_{ij} \tag{1.8a}$$

$$s.t.\ x_{ij} + x_{jk} - x_{ik} \leq 1, \qquad 1 \leq i < j < k \leq n \tag{1.8b}$$

$$x_{ij} - x_{jk} + x_{ik} \leq 1, \qquad 1 \leq i < j < k \leq n \tag{1.8c}$$

$$-x_{ij} + x_{jk} + x_{ik} \leq 1, \qquad 1 \leq i < j < k \leq n \tag{1.8d}$$

$$x_{ij} \in \{0, 1\}, \qquad 1 \leq i < j \leq n \tag{1.8e}$$

In (Grötschel and Wakabayashi, 1989) it is proven that the polyhedron of this problem is the same as the one in Clique Partition problem . This problem is NP-complete and it is limited to solve data samples no more than 300 (Sukegawa et al., 2012). So we will use another method for representing the consensus clustering which is graph illustration (Lancichinetti and Fortunato, 2012).

A simple way to show how the consensus clustering works is graph visualization. Assume that we have 7 data points which are presented by nodes on the graph in figure1.3 and we have four different clusterings of these data points. Each clustering is shown in a circle in figure1.3, if there is an edge between two nodes it means that they are in the same cluster. There are different ways to construct the consensus, here we construct the consensus by giving weights to each of the edges in the consensus graph. The weights can be defined as number of times two nodes on each edge end points appeared in the same cluster divided by the number of total clusterings available. If the weight is higher then it is more likely that the two points are in the same cluster so the edge would be thicker. At the end the consensus graph will be constructed as shown in the figure. Through this process we can construct the adjacency matrix of the graph which will create our consensus matrix. For the illustrated graph the matrix is as follows:

$$
\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}
\begin{array}{c}
\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
\left(\begin{array}{ccccccc}
1/4 & 2/4 & 4/4 & 1/4 & 0 & 0 & 0 \\
2/4 & 1/4 & 2/4 & 3/4 & 1/4 & 0 & 0 \\
4/4 & 2/4 & 1/4 & 1/4 & 0 & 0 & 0 \\
1/4 & 3/4 & 1/4 & 1/4 & 1/4 & 0 & 0 \\
0 & 1/4 & 0 & 1/4 & 1/4 & 3/4 & 3/4 \\
0 & 0 & 0 & 0 & 3/4 & 1/4 & 4/4 \\
0 & 0 & 0 & 0 & 3/4 & 4/4 & 1/4
\end{array}\right)
\end{array}
$$

Each element a_{ij} in consensus matrix is showing the weight between node i and node j which is the number of times these two points showed up in the same cluster out of 4. So the highest weight is $4/4$ which has the thickest edge and assures us that these two points are in the same cluster and the lowest weight is 0 which assures us that these two points are not in the same cluster.

As we can see, by using each of the clusterings alone we will not give any good answers, but when we use consensus clustering, it gives us very useful informations about the data even though some clusterings used in the function have very bad quality.

Our contribution in this thesis is to give a consensus clustering method for time series data based on the definitions we gave above and evaluate the results using clustering evaluation methods. We will use five clustering algorithms for our consensus and we will show how significantly the results can be improved with the help of consensus.

In the following section we will discuss about our methodology and we will describe the clustering algorithms we will use for the consensus clustering.

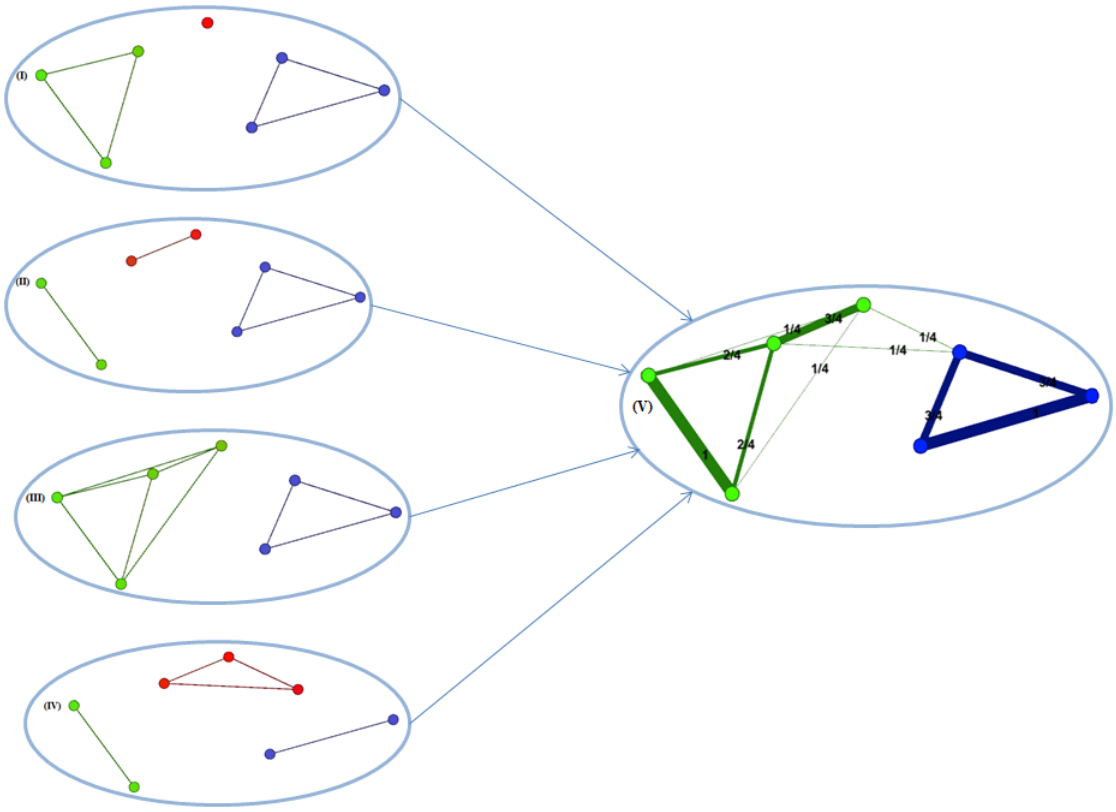


Figure 1.3: consensus clustering graph illustration, (I), (II), (III), (IV) are four different clusterings and (V) is the consensus created based on them.

CHAPTER 2: METHODOLOGY

As discussed in section 2 of the previous chapter, for constructing our consensus matrix we will use five different clustering algorithms including: Hierarchical, K-means, Fuzzy and two spectral clustering algorithms. Each of these algorithms give different clusters. Some of these algorithms give the same results after several runs but some of them don't, like k-means. In this methodology we will not deal with several runs of a same algorithm on the same data, but we will run these five algorithms on the same data once and by using the concept of consensus we will improve the results significantly. This approach will help us to eliminate weak clusterings and outliers and will reveal the structure of the data better than individual algorithms.

Assume that a time series is represented by $D = d_1, d_2, ..., d_n$ and each d_i is a vector of w number of variables(window length). Each clustering algorithm gives labels 1 or 2 to each d_i since we only need two clusters(normal or abnormal). Figure2.1 is a representation of a time series. Each d_i presents a raw in the time series and they each have a certain pattern. Number of columns is the same as window length or number of features. Our problem is a binary classification problem in which we are interested in two classes of balanced or imbalanced. The clusters are not balanced, which means one class might be bigger or smaller than the other class and they are not the same size.

After getting the results of different clusterings we will construct the consensus matrix. Consensus matrix A is a $n \times n$ matrix, and each element has a value between 0 to 1. For each a_{ij}, the value shows how many times out of 5, d_i was in the same cluster as d_j.

For decision making and evaluation purposes through voting, we decide if two data are in the same cluster or not. If the value of a_{ij} is greater than 0.4, which means two points appeared in the same

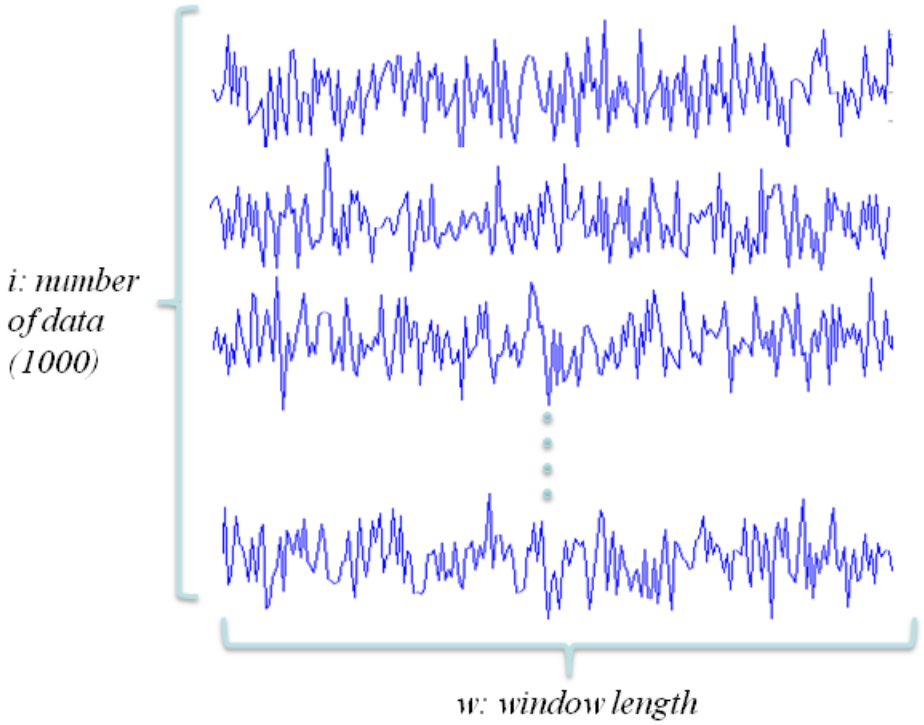


Figure 2.1: Time Series Representation

cluster more than two times, then d_i and d_j are in the same cluster, and if the value is smaller or equal than 0.4 then d_i and d_j are not in the same cluster. In the following sections we will give a brief description of the clustering algorithms we used and then we will evaluate the results.

Clustering Algorithms

K-Means

K-means is a partitioning method first proposed by MacQueen et al. (1967). Given a set of n unlabeled data, a partitioning method creates K partitions of data, where each partition represents a

cluster containing at least one object and $K \leq n$. In K-means, partitions are crisp meaning that each object belongs to exactly one cluster, and each cluster is represented by the mean value of the objects in the cluster.

The goal of k-means is to minimize the objective function which is the total distance between all objects from their respective cluster centers. K-means is an iterative algorithm starts by arbitrarily choosing initial centers for clusters and then assigning objects to the closest cluster centers and updating the clusters. This process continues until the value of the objective function is minimized.

Let the set of data points be represented as $D = \{d_1, d_2, d_3, ..., d_n\}$ where each $d_i = \{d_{i1}, d_{i2}, d_{i3}, ..., d_{iw}\}$ is a vector with w features. Then k-means partitions D into K partitions $P_1, P_2, P_3, ..., P_k$ where the within-cluster sum of squares is minimized:

$$min \sum_{i=1}^{k} \sum_{d_j \in P_j} u_{ki} \|d_j - \mu_j\|^2 \tag{2.1}$$

$\mu_j = Mean\ of\ points\ in\ P_j$ and $u_{ki} \in \{0, 1\} \forall k, i\ and\ \sum_{k=1}^{K} u_{ki} = 1$
In above formula we use euclidean distance for similarity measures but other distance measures can be used in k-means. The algorithm starts by choosing initial centers and assigning points to the closest center. Then the algorithm will revise the center and update the clusters so that the value of the objective function will be minimized, this process will be repeated until the value can't be reduced more than a ε.

In k-means, number of clusters must be defined by user and if the initial centers are not defined by user then they will be assigned by the algorithm. With various numbers of clusters and initial centers, the algorithm will give different results. Also, the algorithm is sensitive to outliers and outliers will effect the result of the clustering.

K-means has been developed in different ways for different applications. Some variations try to

find good initial centers and some of them allow the merging and splitting of resulting clusters, Ball and Hall used this method for ISO data (Ball and Hall, 1965). Euclidean k-medians which is a variation of k-means is presented by Arora et al. (1998).

Hierarchical

Hierarchical clustering works by clustering data into a hierarchy tree. A hierarchical clustering results in a dendrogram representing the clusterings and similarity levels, and by breaking the dendrogram in different levels, we will obtain different numbers of clusterings. There are two types of hierarchical clustering: Agglomerative (bottom up) and Divisive (top down). Agglomerative methods start by placing the objects in their own clusters and then merges clusters into larger clusters until it reaches a single cluster or a certain termination criteria. Divisive methods work the reverse way.

Hierarchical clustering has the problem of adjusting once a merging decision is made. Different measures to calculate the distance between clusters results in different variations of hierarchical clustering such as single link, complete link and minimum variance. In single link, the distance between two clusters is defined as the distance between their two closest points, while in complete link distance of clusters is defined as distance between their two farthest points which makes it sensitive to outliers. Different distance measures and linkage criteria can be use for hierarchical clustering, regarding our data, we used agglomerative because they are more popular, euclidean distance measure and single linkage method, because we want to prevent the effects of outliers and usually single link algorithms are more versatile than complete links (Jain et al., 1999). Ward gave some hierarchical clustering schemes in his paper (Ward Jr, 1963). F. Corpet used hierarchical clustering for an algorithm for the multiple alignment of sequences, either proteins or nucleic acids

(Corpet, 1988). Bandyopadhyay and Coyle proposed a distributed, randomized hierarchical clustering algorithm to organize the sensors in a wireless sensor network into clusters (Bandyopadhyay and Coyle, 2003).

Fuzzy

In fuzzy clustering, partitions are not crisp, meaning that one object can belong to more than one cluster to a different degree. The notation of fuzzy sets first proposed by Zadeh (1965) and the use of fuzzy sets in clustering was first proposed in (Bellman et al., 1964). Fuzzy clustering associates each pattern to a cluster with a membership function. Two commonly used fuzzy algorithms are the fuzzy c-means (Bezdek, 1981) which works better than k-means but they converge to local minima of the squared error criterion (Jain et al., 1999) and the fuzzy c-medoids algorithm (Krishnapuram et al., 2001). These two also work well for finding spherical-shaped clusters and small to medium data sets, but for non-spherical or complex shape data sets, algorithms such as adaptive fuzzy clustering (Krishnapuram and Kim, 1999) works better. A very important part of fuzzy clustering is how to define the membership function. A fuzzy clustering algorithm works generally with these steps: First, construct $M(n \times k)$ membership matrix, n is the number of data and k is the number of clusters. Each element m_{ij} in M shows the degree of membership of data d_i to cluster p_j which is between 0 and 1. Then, using M, create the fuzzy membership function and calculate its value. Reassign data to clusters so that the value of the fuzzy function is decreased and repeat this until the value is minimum or it can't be changed.

Different membership functions can be defined like weighted squared error. One membership

function can be defined as below:

$$\sum_{i=1}^{n}\sum_{j=1}^{k} m_{ij}\|d_i - \mu_k\|^2 \tag{2.2}$$

Where $\mu_k = \sum_{i=1}^{n} m_{ik} d_i$ is the kth fuzzy cluster center. Bezdek discussed the applicability of the fuzzy ISODATA clustering algorithms for dimensionality reduction of binary valued data sets, and computerized medical diagnosis (Bezdek, 1976) and fuzzy clustering is also used for cardiovascular investigations (Kalmanson and Stegall, 1975).

Spectral Clustering

Spectral clustering techniques make use of eigenvalue of the similarity matrix. Spectral clustering goes back to Donath and Hoffman (1973) when they suggested the use of eigenvectors for graph partitions. In this paper we used two spectral clustering algorithms, one proposed by Shi and Malik (2000) for solving the perceptual grouping problem in vision. This algorithm partitions the data into two points based on the normalized Laplacian matrix. Assume that data is represented as $D = \{d_1, d_2, ..., d_n\}$ and pairwise similarity is defined as $s_{ij} = s(d_i, d_j)$ which is measured by a similarity function and the similarity matrix is defined as $S = (s_{ij})_{i,j=1,...n}$ and we are looking for k clusters. First the algorithm will create a similarity graph and W as the weighted adjacency matrix, then unnormalized graph Laplacian L will be calculated. Then the first k eigenvectors $U = \{u_1, u_2, ..., u_k\}$ will be calculated by solving the system $Lu = \lambda Du$. Then we will use a clustering algorithm like k-means to cluster U into k clusters.

The second algorithm is proposed by Ng et al. (2002) which is as follows: let the set of data be represented as $D = \{d_1, d_2, d_3, ..., d_n\}$ and K numbers of clusters and pairwise similarity is defined

as $s_{ij} = s(d_i, d_j)$ which is measured by a similarity function and the similarity matrix is defined as $S = (s_{ij})_{i,j=1,...n}$, at the beginning the algorithm works the same as Shi-Malik algorithm, when we get matrix U we form another matrix T by normalizing the rows of U to norm 1 that is : $t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2}$. Then we will do the k-means clustering on T instead of U.

An important matter in using spectral clustering is to define a good similarity graph, Laplacian matrix and number of clusters. Von Luxburg (2007) defines different graph Laplacians and spectral clustering algorithms and his paper is a good reference for different spectral clustering algorithms.

Spectral clustering also tries to create balance clusters. In binary classification problem, the clusters are not balanced, so in this study we will also see the effects of a bad clustering on consensus.

Evaluation Methods

We used this algorithm for time series data. In this section we are going to show how significant the results of clustering can be improved by the use of consensus.

Different evaluation methods can be used for imbalanced data. The evaluation method we use is based on the confusion matrix. In this data set, data is in either in Positive or Negative class and the algorithms decide on that. So, there are four types of data, if data is positive and algorithm identifies them as positive then it is true positive but if the algorithm identifies them as negative then it is false negative and the same thing applies for negative class, base on this we form the confusion matrix as shown in table2.1.

Where, TP, FN, FP, TN stands for True Positive, False Negative, False Positive and True Negative accordingly. We would like to see higher values on the diagonal of the matrix which shows how

Table 2.1: Confusion matrix for binary classification problem.

Data Identified as	Positive	Negative
Positive Data	TP	FN
Negative Data	FP	TN

good the algorithm can identify data correctly.

For measuring the performance of the clustering algorithm two measures of Sensitivity and Specificity are defined as below:

$$Sensitivity = \frac{TP}{TP + FP} \quad Specificity = \frac{TN}{TN + FN} \tag{2.3}$$

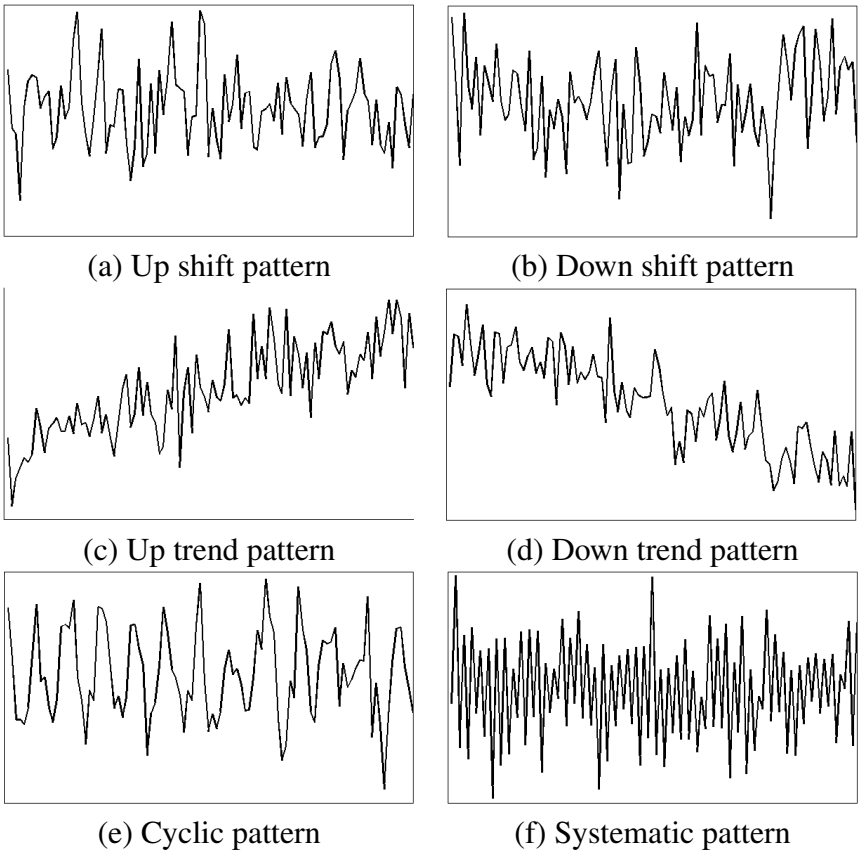
Sensitivity shows how well the algorithm performs on balance data and specificity shows how well the algorithm performs on imbalance data. Geometric mean (G-mean) of sensitivity and specificity is also defined as $\sqrt{Specificity \times Sensitivity}$ which is a good evaluation measure for the performance of the algorithm both for positive and negative class.

We used the defined performance measures to evaluate our proposed method and examine if the results will be improved.

There are other criteria for clustering such as Run Length, Time and Robustness which will be discussed in Results section.

Control Chart Pattern Recognition

In manufacturing, time series pattern recognition is important since slide alterations might be indicative of a malfunction that requires a course of appropriate corrective actions (e.g. maintenance). Manual monitoring requires is tedious and requires person ell's undistracted attention. For this machine learning based automated algorithms, termed Control Chart Pattern Recognition (CCPR) algorithms, have been proposed to detect abnormal behaviors. The term was originally coined by Shewhart (1931). In an early publication of Western Electric Company () totally 15 abnormal patterns were identified figure2.2.



(a) Up shift pattern (b) Down shift pattern

(c) Up trend pattern (d) Down trend pattern

(e) Cyclic pattern (f) Systematic pattern

Figure 2.2: Examples of six basic abnormal patterns

During several years, different pattern recognition algorithms have been studied in literature and a complete literature review of CCPRs is done by Hachicha and Ghorbel (2012). Several supervised and unsupervised methods are used to handle CCPRs including Neural Networks, Correlation Analysis, Principle Component Analysis (PCA) and Time Series Modeling. Neural Networks have been applied to detect abnormal patterns, Guh and Hsieh (1999) proposes an artificial neural network based model, which contains several back propagation networks, to both recognize the abnormal control chart patterns and estimate the parameters of abnormal patterns. Perry et al. (2001) used back-propagation artificial neural networks to model and identify abnormal patterns. A hybrid approach named Hybrid Artificial Neural Network- Naive Bayes classifier is also identified by Adam et al. (2011). Also, BESDOK and ERLER (2000) identified a neural network for speeding up the training process.

Statistical classifications are also used for CCPRs. Yang and Yang (2005) presented a control chart pattern recognition system using a statistical correlation coefficient. Yang and Shahabi (2004) proposed a similarity measure for multivariate time series datasets, Extended Frobenius Norm(Eros), which is based on PCA.

Time series clustering is also a very common method for pattern recognition.

A time series is a sequence of data measured in specific points in time. A time series gives a very useful information about the data and with the help of a time series, different patterns can be identified.

Little attention is paid to clustering of CCPR for imbalanced and balanced data and number of works done on this matter are very few. The first unsupervised approach was proposed by Al-Ghanim (1997) which proposed an unsupervised self-organizing neural paradigm. Al-Ghanim and Kamat (1995) presented unnatural pattern recognition technique using correlation analysis on Trend, Systematic and Cyclic patterns and presented results with evaluation methods. Wang and

Kuo (2007) used three different fuzzy clusterings on CCPR for six patterns and compared their performance. In another article (Wang et al., 2009) Independent Component Analysis (ICA) and Decision Trees is proposed to identifying different patterns. Maximum Likelihood method is proposed by Naeini et al. (2011) and they used confusion matrix for evaluations. Due to a few number of works done on this matter, there is a need for more research and study.

In this study, we used the proposed methodology which is an unsupervised consensus framework on CCPR data.

In the next chapter, we will present our results on CCPR data based on different clustering criterion.

CHAPTER 3: RESULTS

In this section, the result of the given algorithm for imbalance data is given based on the evaluation methods defined in chapter 2, section 2. The algorithm is coded in MATLAB.

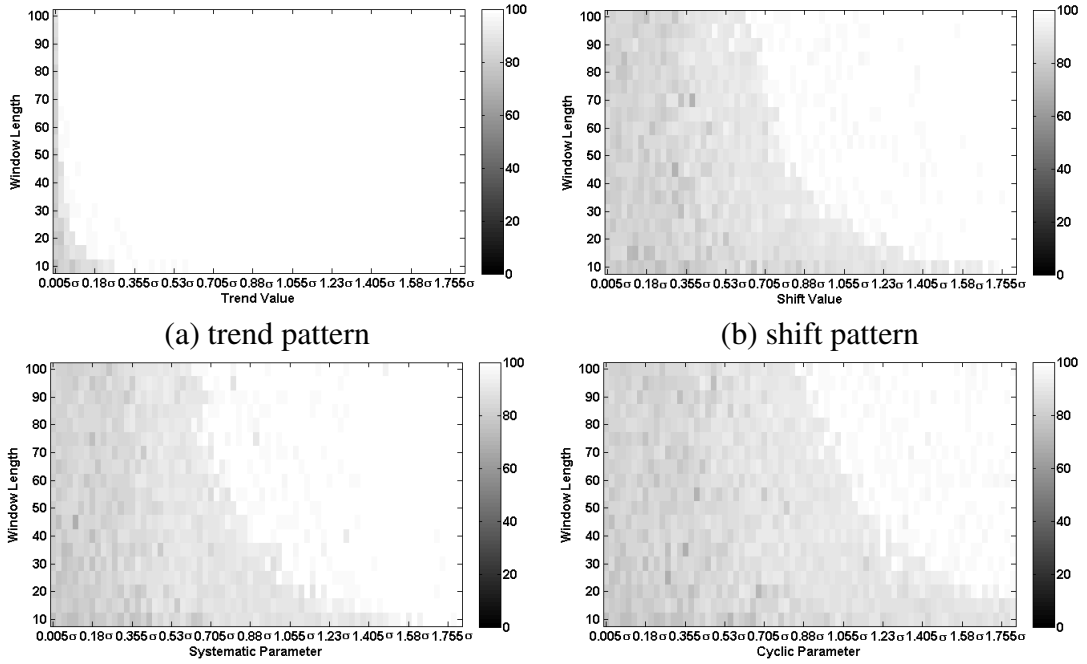
We used simulated CCPR data. Each data has a certain number of features. For each pattern we have a pattern parameter which shows how intense the pattern is, so for higher pattern parameters it is easier for clusterings to detect the pattern. We used a data generator code to generate different patterns with different window length and pattern parameters.

For evaluation purposes we gave labels to our data so later we could compare them with the labels we get from clusterings. Since we are using an unsupervised method, we don't give the labels as an input to the algorithms, but we just have them for evaluation purposes.

CCPR data play an important roll in today's industry since the tasks on big data sets can not be done by humans. The four mentioned patterns are the most common studied in literature. By applying this method to CCPR data we will use the variability and uncertainty existing in today's problems in order to give better results and improve different tasks such as quality control, pattern detection or error detection.

In figure3.1 the g-mean results for consensus matrix for different patterns are given. For each pattern, g-mean is calculated for different window lengths and pattern parameters.

In the chart, the horizontal axis is the pattern parameter which shows how intense the pattern is, so it is easier to detect patterns with high parameters. The vertical axis is the window length or number of features.



(a) trend pattern (b) shift pattern

(c) systematic pattern (d) cyclic pattern

Figure 3.1: consensus clustering results based on g-mean calculations

As the charts in figure3.1 are showing, as we increase the number of windows and clustering parameter, the consensus results improve for all the patterns. As the figure is showing, Trend pattern is easily detected by consensus. Shift pattern is a harder task for the algorithm than Trend, but still the results are very good. Cyclic and Systematic patterns are the hardest to detect both for the consensus and the algorithms, and as the figure is showing, there are more dark areas which means lower g-mean. All in all, the consensus is giving a very good clustering quality since the lowest g-mean for toughest patterns is around 70%.

For each pattern, there exist three states, these states are defined for specific window lengths and pattern parameters for each pattern. These three states include a worst state, a critical state and a good state. Worst state is when different clustering algorithms give weak clusterings and so the consensus, this mainly happens in low pattern parameters and low number of window lengths.

Critical state is were we don't get very good or very bad clusterings from algorithms but the result of the consensus will give us good answers, this happens when window length and pattern parameter make the task hard for some algorithms. Good state is when we will get good results both from the algorithms and the consensus. We would like to mention that in all three states the consensus gives us better results than other algorithms as we will discuss later. For example, as figure3.1 shows, a window length 15 and trend parameter 0.005 is a worst state for trend pattern and window length 90 and trend parameter 1.055 is a good state. For different patterns these states differ, for instance while (80,1.055) is a good state for trend, it is a critical state for cyclic pattern.

In table3.1, we gave a comparison between the consensus clustering results and k-means in three different state of each pattern. In these points there is a major change in the consensus clustering result and they can be good representatives for comparing the results of the consensus with each of the algorithms.

As you can see in the table, the results, both sensitivity and specificity and the overall g-mean is improving dramatically over the consensus and this is because the consensus eliminates weak clusterings and improves the results. Only in some good states we can see that k-means is giving better g-means but they are not a considerable amount.

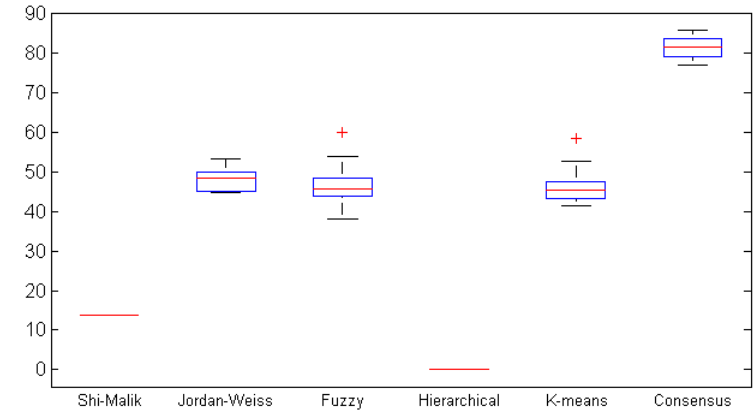
The robustness of the clustering algorithm plays an important roll in some problems, a desirable clustering algorithm should cluster data the same way after several times of running. This is very important for big data sets or when there is a need to run clustering on the data from different categories of individuals or from different sources.A good clustering algorithm should cluster the same type of data the same way after several runs.

For measuring this characteristic of the consensus and comparing it with other algorithms, we used box plots as shown in figure3.2. Using box plots gives information about the degree of dispersion

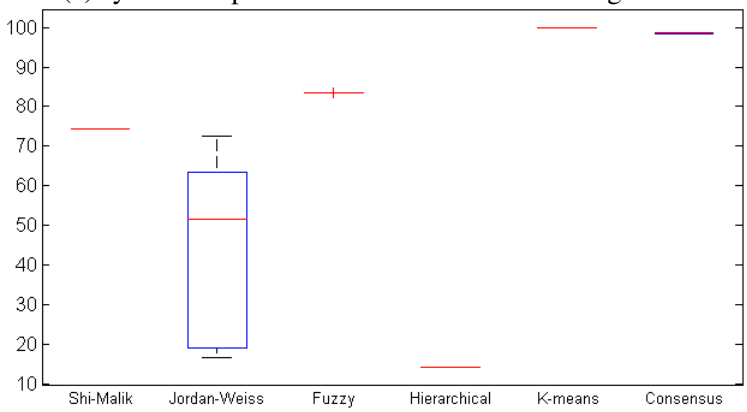
Table 3.1: comparison between k-means clustering and consensus clustering based on sensitivity, specificity and g-mean for different patterns and parameters

	K-means			Consensus			Parameters
Abnormal Pattern	Sen	Spec	G	Sen	Spec	G	(Windows,Parameter)
Uptrend	100	100	100	100	100	100	(60,0.205)
	52.42	92	69.44	86.1	95.91	90.88	(85,0.006)
	50.52	58	54.13	80.02	78.2	79.1	(60,0.004)
Downtrend	100	100	100	100	100	100	(60,0.205)
	44.21	34	38.77	85.15	87.59	86.36	(85,0.006)
	53.57	54	53.78	84.13	81.87	82.99	(60,0.004)
Upshift	99.89	100	99.94	97.81	100	98.9	(60,0.805)
	51.36	22	33.61	83.74	90.53	87.07	(80,0.405)
	55.89	62	58.86	80.87	85.46	83.14	(40,0.205)
Downshift	99.68	100	99.84	99.44	100	99.22	(60,0.805)
	44.94	62	52.78	84.9	96.97	90.73	(80,0.405)
	52.31	30	39.61	82.88	88.73	85.75	(40,0.205)
Cyclic	99.15	100	99.57	96.83	99.34	98.08	(50,1.205)
	57.78	98	75.25	82.75	97.22	89.7	(50,0.805)
	51.89	38	44.4	83.17	79.34	81.24	(40,0.405)
Systematic	99.47	100	99.73	97.74	100	98.86	(60,0.805)
	61.05	100	78.13	83.04	100	91.12	(50,0.605)
	48.52	60	53.95	81.57	80.24	80.9	(20,0.205)

and skewness of the clustering results. For our problem, we run each clustering algorithm 30 times on the same data and at the end we calculate the g-means for each run of the algorithms. Each box plot shows the g-means of each of the algorithm for 30 times of run. As the figure figure3.2.a is showing, Hierarchical clustering and Shi-Malik algorithms are giving the same result each time, but with a very low g-mean.The g-means of other three clusterings is changing between 35% to 60% while consensus results is changing between 80% to 90% which is higher and more robust than the others. Figure figure3.2.b is a less challenging problem, but as we can see, the results of consensus is better than the other algorithms and it is more robust.



(a) systematic parameter=0.18 and window length=20



(b) systematic parameter=1.405 and window length=50

Figure 3.2: robustness of consensus clustering in compare to other clustering algorithms for systematic pattern. (a) is a worst state for systematic parameter and (b) is a good state.

So in conclusion, consensus clustering results are more robust than the other algorithms and it does not change very much over different times of running.

Another important factor in algorithms is the running time, in figure3.3 the running time of the algorithm in seconds is given for different data size. As shown in the figure, running time for data size below 1000 is almost 0 while there is a sudden increase when the data size is increased to 2000 and after that, until we reach 10000 data size which used 2 hours and 51 minutes which is a good time for such a big data. This running time is calculated on a laptop with Intel core i7 CPU and 8

GB of RAM. This running time is also worst case because it is calculated for systematic patterns which are the hardest to cluster. In industry this running time can be decreased to almost zero by the use of different computers at the same time, because we are using different clusterings, we can run each clustering in a different computer at the same time and get the results from each of them, in this way the running time will be decreased significantly.

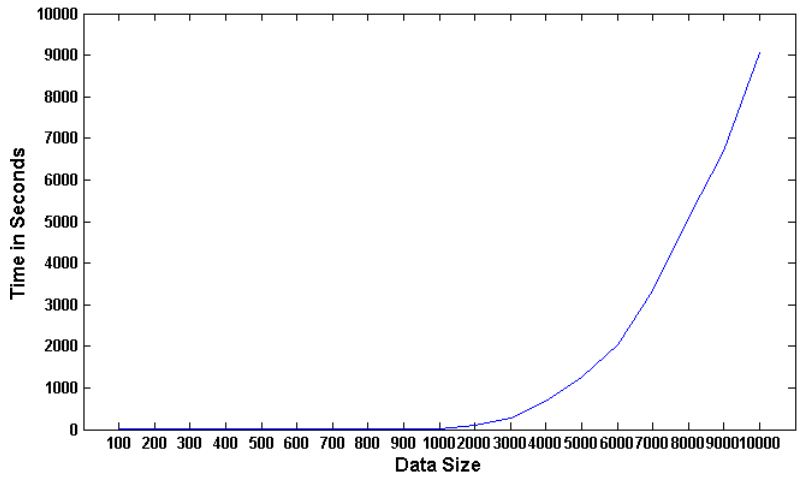


Figure 3.3: Running time of consensus clustering in seconds for different data sizes.

Another measurement is the average run length (ARL) which evaluates the speed of pattern recognizer. ARL is the average number of observations required before the expected pattern is detected Hwarng and Hubele (1992). ARL is calculated from those sequences where the expected pattern is recognized by a certain run length and we exclude the sequences where the pattern was not recognized at all. So the ARL will be divided by the percentage of detected patterns in order to exclude the undetected ones and this parameter is called average run length index (ARLIDX).

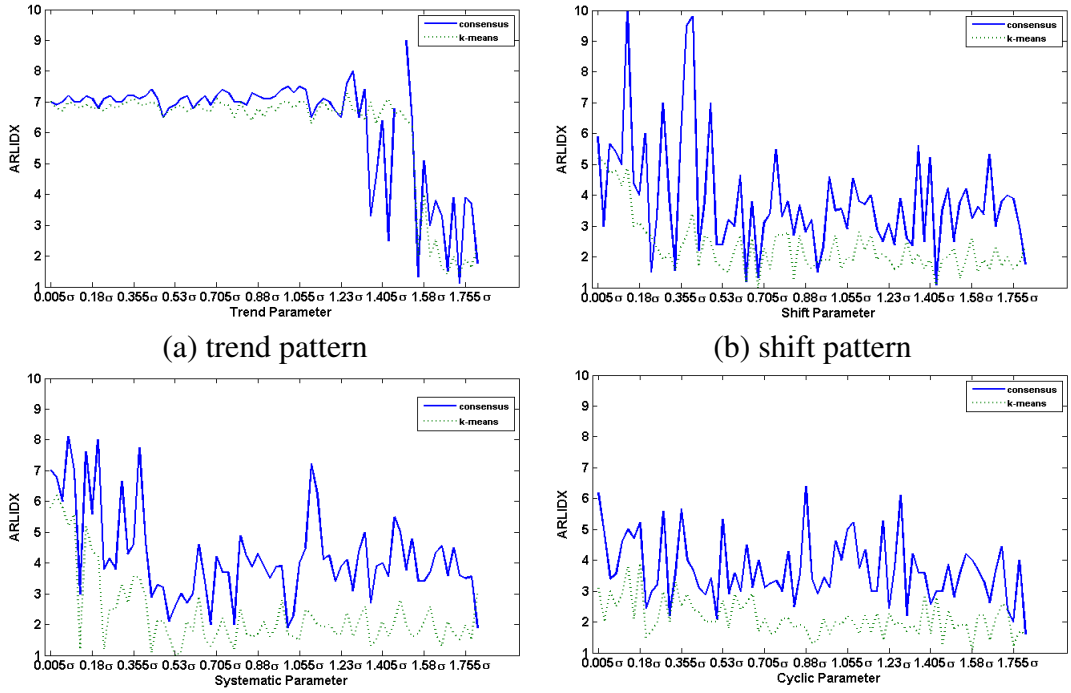
ARLIDX will give us information about how long it takes for a clustering algorithm to detect an abnormal pattern, in this method, we will add an abnormal data to another data and run the clus-

tering, if the data is detected as abnormal then the run length is 1, if not we will keep adding more abnormal data until the pattern is detected.

In figure3.4 we compared ARLIDX between k-means and consensus for different patterns with different clustering parameters but with the same window length. As the figure shows, k-means gives a better ARLIDX than consensus in most of the cases, but the difference is not very significant. This is for 10 window length which is very challenging for clustering algorithms, and other algorithms will affect the results of consensus so they bring the ARLIDX higher. As shown in table3.2, consensus performs better as in ARLIDX than most of the clusterings, and except two cases, the detection percentage is 100%, which means all the patterns are detected.

By looking at figure3.4 and table3.2, we can see that the ARLIDX of consensus is affected by weak clusterings but not significantly, although the detection ratio is not so much affected.

If we add table3.3 to our observations, as the window length increases and the problem becomes less challenging, consensus outperforms k-means significantly, until in 300 window length there is 30 difference! With all of the observations, we can conclude that in challenging situations (window length 10 and below) k-means out performs the consensus in case of ARLIDX, and consensus outperforms the other clusterings, but after a certain window length consensus outperforms all of the algorithms significantly.



(a) trend pattern (b) shift pattern

(c) systematic pattern (d) cyclic pattern

Figure 3.4: Comparison of Average Run Length Index between consensus and k-means for different patterns with window length 10.

Table 3.2: Average Run Length Index for different parameters and window lengths for various patterns

	Shi-Malik		Jordan-Weiss		Fuzzy		Hierarchical		K-Means		Consensus		Parameters
Abnormal Pattern	ARLIDX	%	ARLIDX	%	ARLIDX	%	ARLIDX	%	ARLIDX	%	ARLIDX	%	(Windows,Parameter)
Trend	35.14	70	1.9	100	39.7	100	40	100	39.4	100	39.9	100	(60,0.205)
	31.7	100	2.3	100	2.8	100	78	10	2.4	100	8.2	100	(85,0.006)
	14.5	100	1.8	100	2.4	100	6	10	2.1	100	5.1	100	(60,0.004)
Shift	28.83	60	1.9	100	3.8	90	18	10	27.5	100	43	30	(60,0.805)
	16.2	100	1.4	100	1.6	100	INFT	0	1.7	100	3.3	100	(80,0.405)
	10.5	90	2.8	100	1.9	100	INFT	0	2.5	100	9.4	100	(40,0.205)
Cyclic	23	100	2.2	100	2.6	100	INFT	0	16.7	100	22.6	100	(50,1.205)
	8.2	100	1.8	100	1.7	100	39.25	40	2.4	100	8.4	100	(50,0.805)
	11	100	1.6	100	1.9	100	INFT	0	2.1	100	8.3	100	(40,0.405)
Systematic	12.3	100	2.7	100	1.1	100	34.75	40	25.7	100	27.62	80	(60,0.805)
	13.4	100	1.4	100	2.2	100	INFT	0	3	100	7.9	100	(50,0.605)
	11.66	30	1.6	100	2	100	INFT	0	1.8	100	3	100	(20,0.205)

Table 3.3: ARLIDX comparison by the change of window length.

	Consensus	K-Means	Parameters
Abnormal Pattern	ARLIDX	ARLIDX	(Windows,Parameter)
Systematic	1.3	1.1	(25,0.805)
	11.4	6.2	(50,0.805)
	2.3	42.2	(100,0.805)
	63.9	88.6	(200,0.805)
	116.6	146.4	(300,0.805)

In the next table, we are giving another comparison between the algorithms. As you can see in table3.4, the first two algorithms which are the spectral clusterings are giving bad ARLIDX and their answer is changing for different runs while the others together with consensus have similar robust answers.

The reason for this is that in spectral clustering, the algorithms try to make balanced clusters while in our problem the clusters are imbalanced. So here we can see that even by using bad clusterings, the results of consensus will still outperform the others and will be satisfying.

Table 3.4: Average Run Length Index for same parameters and window length for several times of running

Parameters	Shi-Malik	Jordan-Weiss	Fuzzy	Hierarchical	K-Means	Consensus
(Windows,Parameters)	ARLIDX	ARLIDX	ARLIDX	ARLIDX	ARLIDX	ARLIDX
(0,9,20)	9	2.3	13.4	13.9	13.2	13.7
(0,9,20)	14	1.6	13.5	14.1	13.5	13.7
(0,9,20)	12.1	2.1	13.5	14.1	13.2	13.6
(0,9,20)	15.2	1.7	13.2	13.7	13.1	13.6

CHAPTER 4: CONCLUSION

In this study, we applied an unsupervised consensus framework for CCPR data which uses five clustering algorithms. As discussed, the consensus improves the results of the clusterings in different aspects such as robustness and good clusters. The use of consensus helps us to find the similarities and eliminate weak clusterings. This is very useful when we are dealing with a large data set and when the other clustering algorithms can't give us good results, also if we are looking for a good and robust clusters after several times of clustering of the same data, consensus clustering is a good option.

As was discussed, consensus clustering gives us the option to do clustering in a very fast way by using different computers working at the same time on different algorithms and at the end one computer can gather the results and generate the consensus.

Also we show that even if we have some bad clusterings, the consensus will still generate good results and this is very critical in unsupervised learning since we do not have much information about data and we do not know which algorithms are best for our problem. So by use of consensus method we can use any kind of clustering on our data and be sure that the results of consensus will be of a good quality independent of other clusterings.

The application of consensus clustering can be very useful in CCPR data. In real world, most of the times there are not enough information about the data and there are lots of uncertainties on the number of clusters or the appropriate clusterings, so different methods are used in order to generate different results. By using consensus we can use this variability and give better and more stable results.

There are several adjustments that can be applied to this consensus algorithm for future work, one

of them is to give weights to each clustering algorithm based on the clustering quality related to them, so if we know that one algorithm gives better results then we can assign more weights to it to improve the results, in this way, clusterings that are more efficient for a certain data type have more weights and they effect the results more than the others, but this requires a thorough study of different clusterings and their weaknesses. Another way to apply weights is to run one algorithm more than one time so to increase its effect.

There is also a need to study different clusterings for specific data types and examine them in order to find appropriate algorithms to use for consensus.

LIST OF REFERENCES

Adam, A., Chew, L. C., Shapiai, M. I., Jau, L. W., Ibrahim, Z., and Khalid, M. (2011). A hybrid artificial neural network-naive bayes for solving imbalanced dataset problems in semiconductor manufacturing test process. In *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on*, pages 133–138. IEEE.

Adey, W. (1972). Organization of brain tissue: Is the brain a noisy processor? *International Journal of Neuroscience*, 3(6):271–284.

Al-Ghanim, A. (1997). An unsupervised learning neural algorithm for identifying process behavior on control charts and a comparison with supervised learning approaches. *Computers & industrial engineering*, 32(3):627–639.

Al-Ghanim, A. M. and Kamat, S. J. (1995). Unnatural pattern recognition on control charts using correlation analysis techniques. *Computers & Industrial Engineering*, 29(1):43–47.

Arora, S., Raghavan, P., and Rao, S. (1998). Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 106–113. ACM.

Ball, G. H. and Hall, D. J. (1965). Isodata, a novel method of data analysis and pattern classification. Technical report, DTIC Document.

Bandyopadhyay, S. and Coyle, E. J. (2003). An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1713–1723. IEEE.

Bellman, R., Kalaba, R., and Zadeh, L. (1966). Abstraction and pattern classification. *Journal of Mathematical Analysis and Applications*, 13(1):1–7.

Bellman, R., Kalaba, R., and Zadeh, L. A. (1964). Abstraction and pattern classification. Technical report, DTIC Document.

BESDOK, E. and ERLER, M. (2000). Control chart pattern recognition using artificial neural networks. *Turk J Elec Engin*, 8(2).

Bezdek, J. C. (1976). Feature selection for binary data: Medical diagnosis with fuzzy sets. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*, pages 1057–1068. ACM.

Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Kluwer Academic Publishers.

Corpet, F. (1988). Multiple sequence alignment with hierarchical clustering. *Nucleic acids research*, 16(22):10881–10890.

Donath, W. E. and Hoffman, A. J. (1973). Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425.

Dubes, R. C. (1987). How many clusters are best?-an experiment. *Pattern Recognition*, 20(6):645–663.

Ghaemi, R., Sulaiman, M. N., Ibrahim, H., and Mustapha, N. (2009). A survey: clustering ensembles techniques. *World Academy of Science, Engineering and Technology*, 50:636–645.

Goder, A. and Filkov, V. (2008). Consensus clustering algorithms: Comparison and refinement. In *ALENEX08: Procs. 10th Workshop on Algorithm Enginering and Experiments*, pages 109–117.

Grötschel, M. and Wakabayashi, Y. (1989). A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45(1-3):59–96.

Guh, R.-S. and Hsieh, Y.-C. (1999). A neural network based model for abnormal pattern recognition of control charts. *Computers & Industrial Engineering*, 36(1):97–108.

Gustafson, D. E. and Kessel, W. C. (1978). Fuzzy clustering with a fuzzy covariance matrix. In *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, volume 17, pages 761–766. IEEE.

Hachicha, W. and Ghorbel, A. (2012). A survey of control-chart pattern-recognition literature (1991-2010) based on a new conceptual classification scheme. *Computers & Industrial Engineering*.

Han, J., Kamber, M., and Pei, J. (2006). *Data mining: concepts and techniques*. Morgan kaufmann.

Hwarng, H. B. and Hubele, N. F. (1992). Boltzmann machines that learn to recognize patterns on control charts. *Statistics and computing*, 2(4):191–202.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Kalmanson, D. and Stegall, H. F. (1975). Cardiovascular investigations and fuzzy sets theory. *The American Journal of Cardiology*, 35(1):80–84.

Krishnapuram, R., Joshi, A., Nasraoui, O., and Yi, L. (2001). Low-complexity fuzzy relational clustering algorithms for web mining. *Fuzzy Systems, IEEE Transactions on*, 9(4):595–607.

Krishnapuram, R. and Kim, J. (1999). A note on the gustafson-kessel and adaptive fuzzy clustering algorithms. *Fuzzy Systems, IEEE Transactions on*, 7(4):453–461.

Kuncheva, L., Hadjitodorov, S., and Todorova, L. (2006). Experimental comparison of cluster ensemble methods. In *Information Fusion, 2006 9th International Conference on*, pages 1–7. IEEE.

Lancichinetti, A. and Fortunato, S. (2012). Consensus clustering in complex networks. *Scientific reports*, 2.

Li, T. and Ding, C. (2008). Weighted consensus clustering. *Mij*, 1(2).

Li, T., Ding, C., and Jordan, M. I. (2007). Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 577–582. IEEE.

Li, T., Ogihara, M., and Ma, S. (2004). On combining multiple clusterings. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 294–303. ACM.

Lin, J., Vlachos, M., Keogh, E., and Gunopulos, D. (2004). Iterative incremental clustering of time series. In *Advances in Database Technology-EDBT 2004*, pages 106–122. Springer.

Liu, Y., Li, Z., Xiong, H., Gao, X., and Wu, J. (2010). Understanding of internal clustering validation measures. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 911–916. IEEE.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA.

Naeini, M. K., Owlia, M., and Fallahnezhad, M. (2011). A new statistical method for recognition of control chart patterns. In *Quality and Reliability (ICQR), 2011 IEEE International Conference on*, pages 609–612. IEEE.

Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856.

Oates, T., Firoiu, L., and Cohen, P. R. (1999). Clustering time series with hidden markov models and dynamic time warping. In *Proceedings of the IJCAI-99 workshop on neural, symbolic and reinforcement learning methods for sequence learning*, pages 17–21. Citeseer.

Perry, M. B., Spoerre, J. K., and Velasco, T. (2001). Control chart pattern recognition using back propagation artificial neural networks. *International Journal of Production Research*, 39(15):3399–3418.

Shewhart, W. A. (1931). Economic control of quality of manufactured product. *New York*, 501.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905.

Sukegawa, N., Yamamoto, Y., and Zhang, L. (2012). Lagrangian relaxation and pegging test for the clique partitioning problem. *Advances in Data Analysis and Classification*, pages 1–29.

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

Verma, D. and Meila, M. (2003). A comparison of spectral clustering algorithms.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

Wang, C.-H., Dong, T.-P., and Kuo, W. (2009). A hybrid approach for identification of concurrent control chart patterns. *Journal of Intelligent Manufacturing*, 20(4):409–419.

Wang, C.-H. and Kuo, W. (2007). Identification of control chart patterns using wavelet filtering and robust fuzzy clustering. *Journal of Intelligent Manufacturing*, 18(3):343–350.

Wang, X., Smith, K., and Hyndman, R. (2006). Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery*, 13(3):335–364.

Wang, X., Smith, K. A., Hyndman, R., and Alahakoon, D. (2004). A scalable method for time series clustering. *Unrefereed research papers*, 1.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Warren Liao, T. (2005). Clustering of time series dataa survey. *Pattern Recognition*, 38(11):1857–1874.

Xiong, Y. and Yeung, D.-Y. (2004). Time series clustering with arma mixtures. *Pattern Recognition*, 37(8):1675–1689.

Y, A., Jordan, M. I., and Weiss, Ng, Y. (2001). On spectral clustering1 analysis and an algorithm. *Proceedings of Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press*, 14:849–856.

Yang, J.-H. and Yang, M.-S. (2005). A control chart pattern recognition system using a statistical correlation coefficient method. *Computers & Industrial Engineering*, 48(2):205–221.

Yang, K. and Shahabi, C. (2004). A pca-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM international workshop on Multimedia databases*, pages 65–74. ACM.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, 8(3):338–353.