CLUSTERING OF LARGE TIME-SERIES DATASETS USING A MULTI-STEP APPROACH

SAEED REZA AGHABOZORGI SAHAF YAZDI

THESIS SUBMITTED IN FULFILMENT
OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2013

UNIVERSITY MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: **SAEED REZA AGHABOZORGI SAHAF YAZDI**

I.C/Passport No: **X95385279**

Registration/Matric No: **WHA080005**

Name of Degree: **DOCTOR OF PHILOSOPHY**

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):
CLUSTERING OF LARGE TIME-SERIES DATASETS USING A MULTI-STEP APPROACH

Field of Study: **DATA MINING**

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date

Subscribed and solemnly declared before,

Witness's Signature Date

Name:

Designation:

ABSTRACT

Various data mining approaches are currently being used to analyse data within different domains. Among all these approaches, clustering is one of the most-used approaches, which is typically adopted in order to group data based on their similarities. The data in various systems such as finance, healthcare, and business, are stored as time-series. Clustering such complex data can discover patterns which have valuable information. Time-series clustering is not only useful as an exploratory technique but also as a subroutine in more complex data mining algorithms. As a result, time-series clustering (as a part of temporal data mining research) has attracted increasing interest for use in various areas such as medicine, biology, finance, economics, and in the Web.

Several studies which focus on time-series clustering have been conducted in said areas. Many of these studies focus on the time complexity of time-series clustering in large datasets and utilize dimensionality reduction approaches and conventional clustering algorithms to address the problem. However, as is the case in many systems, conventional clustering approaches are not practical for time-series data because they are essentially designed for static data and not for time-series data, which leads to poor clustering accuracy. Adequate clustering approaches for time-series are therefore lacking.

In this thesis, the problem of the low quality in existing works is taken into account, and a new multi-step clustering model is proposed. This model facilitates the accurate clustering of time-series datasets and is designed specifically for very large time-series datasets. It overcomes the limitations of conventional clustering algorithms in dealing with time-series data.

In the first step of the model, data is pre-processed, represented by symbolic aggregate approximation, and grouped approximately by a novel approach. Then, the groups are

refined in the second step by using an accurate clustering method, and a representative is defined for each cluster. Finally, the representatives are merged to construct the ultimate clusters. The model is then extended as an interactive model where the results garnered by the user increase in accuracy over time. In this work, the accurate clustering based on shape similarity is performed. It is shown that clustering of time-series does not need to calculate the exact distances/similarity between all time-series in a dataset; instead, by using prototypes of similar time-series, accurate clusters can be obtained.

To evaluate its accuracy, the proposed model is tested extensively by using published time-series datasets from diverse domains. This model is more accurate than any existing work and is also scalable (on large datasets) due to the use of multi-resolution of time-series in different levels of clustering. Moreover, it provides a clear understanding of the domains by its ability to generate hierarchical and arbitrary shape clusters of time-series data.

ACKNOWLEDGEMENT

First, I would like to thank Dr.Teh Ying Wah, with whom I have enjoyed working and who is a great advisor. Thank you for your encouragement, support, and direction you have provided during the past four years. I have been very grateful to have you as a counsellor and advisor.

My special thanks go to my friends in the Faculty of Computer Science and Information Technology (FSKTM) at University of Malaya for the helpful discussions we had about my project. My gratitude also goes to Dr. Keogh for sharing the UCR dataset with me, and guiding me on time-series concepts. I also thank Mr.Zia Madani for provision of the Bank data. I owe a lot to him for his support during my hard days in Malaysia.

I wish to thank my parents for the encouragement and for providing me with so many opportunities to improve in academics. Thank you for the invaluable support you gave me in the course of my life and studies.

Finally, I am deeply grateful to my wife, Mahda, for her ongoing moral support, and acceptance of my long hours away from our family. Thank you so much for your constant encouragement and ridiculous amounts of patience.

Kuala Lumpur, 7th March 2013

Saeed R. Aghabozorgi

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

MTC	Multi-step Time-series Clustering
ED	Euclidean Distance
DTW	Dynamic Time Warping
LCSS	Longest Common Sub-Sequence
SAX	Symbolic Approximate
TS	Time-series
MINDIST	Minimum Distance
PAA	Piecewise Aggregate Approximation
APXDIST	Approximate Distance
IMTC	Interactive Multi-step Time-series Clustering
CAST	Cluster Affinity Search Technique
SOM	Self-Organization Map
FCM	Fuzzy C-Means
DWT	Discrete Wavelet Transformation
PCS	Pure Cluster Search
GT	Grand Truth

1.0 INTRODUCTION

1.1 Background: Time-series Clustering

Data mining is the analysis step of knowledge discovery in database process used to discover new patterns from large datasets (Fayyad, Piatetsky-Shapiro, & Smyth, 1996), and has had a profound impact on our society by solving real-life problems (Chakrabarti et al., 2006). Data mining aims to extract useful knowledge and to summarize it to make it understandable so it can be utilized for further human use (Hand, Mannila, & Smyth, 2001).

Data mining involves different techniques. Clustering is a data mining technique where similar data are placed into related groups (ideally) without advanced knowledge of the groups' definitions. According to Mirkin (2005), clustering is defined as "a discipline devoted to finding and describing cohesive or homogeneous chunks in data, the clusters." Clusters are formed by grouping objects that have maximum similarity with other objects within the group, and minimum similarity with objects in other groups. It is a useful approach for exploratory data analysis as it identifies structure(s) in an unlabelled dataset by objectively organizing data into similar groups. Moreover, clustering is used for exploratory data analysis, for summary generation, and as a pre-processing step for either other data mining tasks or as part of a complex system.

A sequence composed of a series of nominal symbols from a particular alphabet is usually called a temporal sequence, and a sequence of continuous, real-valued elements, is known as a time-series (Antunes & Oliveira, 2001). A time-series is essentially classified as dynamic data because its feature values change as a function of time. That is, the value(s) of each point of a time-series is/are one or more observations made chronologically. Informally, time-series data is a type of temporal data which is naturally high dimensional and large in data size (Keogh & Kasetty, 2003; J. Lin,

Vlachos, Keogh, & Gunopulos, 2004; Rani & Sikka, 2012). The time-series is defined formally as:

Definition 1.1: Time-series, a time-series $F_i = \{f_1, .., f_t, .., f_T\}$ is an ordered set of flow vectors which indicate the spatiotemporal characteristics of moving objects at any time t of the total track life T (Morris & Trivedi, 2009). A flow vector or feature vector $f_t = [X, Y, Z, ...]$ generally represents location or dynamics in a domain. However, just a spatial location $f_t = [X]$ is taken into account in this work because the majority of time series clustering applications are univariate time series (Warrenliao, 2005). It is assumed that $D = \{F_1, .., F_i, .., F_N\}$ is a collection of time-series in a domain, where F_i represents i-th time-series (i = 1,..,n) in the domain.

Time-series data are of interest due to their ubiquity in various areas ranging from science, engineering, business, finance, economics, healthcare, to government (Warrenliao, 2005). Usually, the stored data in these systems are time-series data such as patients' heartbeat electrocardiogram (ECG), daily temperatures, human DNA sequences, weekly sales totals, prices of mutual funds and stocks, moving objects' trajectories, and Web usage sequences (refer to Section 2.2 for references). In addition, considering some data, such as images, text, handwriting, and video, as time-series data may be beneficial (Ratanamahatana & Keogh, 2004a; Ratanamahatana & Niennattrakul, 2006; Ratanamahatana, 2005). Each time-series, while consisting of a large number of data points, can also be seen as a single object (R. Kumar & Nagabhushan, 2006). Clustering such complex objects (time-series data) is particularly advantageous because it leads to the discovery of interesting patterns in the time-series datasets. These patterns can be either frequent or rare patterns. Accordingly, several research challenges have arisen, such as developing methods to recognize dynamic changes in time-series, anomaly and intrusion detection, process control, and character recognition (Chiş, Banerjee, & Hassanien, 2009; Faloutsos, Ranganathan, & Manolopoulos, 1994; X.

Wang, Smith, & Hyndman, 2006). More applications of time-series data are discussed in Section 2.2.1.

With questions such as "Why is clustering of time-series data important?" and "Why would one need to cluster a time-series dataset?", potentially overlapping objectives for clustering of time-series data are given as follows:

1. Time-series databases contain valuable information which can be obtained through pattern discovery. Clustering is a common solution performed to uncover these patterns on time-series datasets.
2. Time-series databases are very large and cannot be handled well by human inspectors. Hence, many users prefer to deal with structured datasets rather than very large datasets. As a result, time-series data are represented as a set of groups of similar time-series by aggregation of data in non-overlapping clusters or by a taxonomy as a hierarchy of abstract concepts.
3. Time-series clustering is the most-used approach as an exploratory technique, and also as a subroutine in more complex data mining algorithms, such as rule discovery, indexing, classification, and anomaly detection (Chiş et al., 2009).
4. Representing time-series cluster structures as visual images (visualization of time-series data) can help users quickly understand the structure of data, clusters, anomalies, and other regularities in datasets.

The problem of clustering of time-series data is formally defined as follows:

Definition 1.2: Time-series clustering, given a dataset of n time-series data $\mathrm{D} = \{\mathrm{F}_1, \mathrm{F}_2, .., \mathrm{F}_\mathrm{n}\}$, the process of unsupervised partitioning of D into $\mathrm{C} = \{\mathrm{C}_1, \mathrm{C}_2, .., \mathrm{C}_\mathrm{k}\}$, in such a way that homogenous time-series are grouped together based on a certain similarity measure, is called time-series clustering. Then, $\mathrm{C_i}$ is called a cluster, where $D = \bigcup_{i=1}^{k} C_i$ and $C_i \cap C_j = \emptyset$ for $i \neq j$.

1.2 Motivation

Data are stored on disks as raw data. Therefore, time-series databases are often very large databases. For example, 1 hour of ECG (electrocardiogram) data needs 1 gigabyte; a typical weblog requires 5 gigabytes per week; the space shuttle database has 200 gigabytes and updating it requires 2 gigabytes per day (Keogh, 2006). This leads to an exponential decrease in the clustering process speed. Additionally, time-series data are often high dimensional (Keogh, Chakrabarti, Pazzani, & Mehrotra, 2001a; J. Lin, Keogh, & Truppel, 2003) which makes handling these data difficult for many clustering algorithms (X. Wang, Smith, Hyndman, & Alahakoon, 2004). It also slows the process of clustering (H. Zhang, Ho, Zhang, & Lin, 2006).

Moreover, to make the clusters, similar time-series should be found. It needs time-series similarity matching (similarity measure), that is, the process of calculating the similarity among the whole time-series. This process is also known as "whole sequence matching" where whole lengths of time-series are considered during distance calculation. However, the process is not simple, because first, time-series data are naturally noisy and include outliers and shifts (J. Lin, Vlachos, et al., 2004); second, the length of time-series varies and the distance among them needs to be calculated. These common issues have made the similarity measure a major challenge for data miners.

Considering all these difficulties in the clustering of time-series, dimensionality reduction is the common solution to increase the performance and speed of the mining process. Dimensionality reduction is a pre-processing action considered to be a fundamental and important process in time-series data mining. It represents the raw time-series in another space by transforming time-series to a lower dimensional space or by feature extraction. But why is dimensionality reduction very important in the clustering of time-series? First, this action is necessary because it reduces memory requirements as all raw time-series cannot fit in the main memory (Keogh & Pazzani,

2000; J. Lin, Keogh, Lonardi, & Chiu, 2003). Second, distance calculation among raw data is computationally expensive, and dimensionality reduction significantly speeds up clustering (Keogh & Pazzani, 2000; J. Lin, Keogh, Lonardi, et al., 2003). Finally, when measuring the distance between two raw time-series, highly unintuitive results may be garnered because some distance measures are very sensitive to some "distortions" in the data (Ratanamahatana, Keogh, Bagnall, & Lonardi, 2005; Ratanamahatana, 2005). That is, by using raw time-series, one may find the clusters of time-series which are similar in noise instead of clusters which are similar in shape. The potential to obtain a different type of cluster is the reason why choosing the appropriate approach for dimension reduction (feature extraction) and its ratio is a challenging task (H. Zhang et al., 2006). In fact, it is a trade-off between speed and quality. All efforts must be made to consider the quality and execution time in the clustering of time-series. Resolving these problems has prompted the researcher to carry out this study to improve time-series clustering.

1.3 Problem Statement

Researchers have shown that generally, clustering by using well-known conventional algorithms such as k-Means, SOM (Self Organization Map), FCM (Fuzzy C-Means), and hierarchy, generate clusters with acceptable structural quality and consistency, and are partially efficient in terms of execution time and accuracy for static data (Jain, Murty, & Flynn, 1999). However, classic machine learning and data mining algorithms do not work well for time-series due to their unique structure (J. Lin, Vlachos, et al., 2004). The high dimensionality, very high feature correlation, and the (typically) large amount of noise that characterize time-series data present a difficult challenge for clustering (Keogh & Kasetty, 2003; J. Lin, Vlachos, et al., 2004). Accordingly, massive research efforts have been made to present methods for time-series clustering. However, focusing on the efficiency and scalability of these algorithms to deal with time-series data has come at the expense of losing the usability and effectiveness of clustering

(Ratanamahatana, 2005). Considering this problem, new approaches for time-series clustering have been proposed to improve not only its efficiency but also the clustering accuracy of time-series data (Bagnall & Janacek, 2005; X. Wang et al., 2006). For example, Ratanamahatana & Niennattrakul (2006) and Niennattrakul & Ratanamahatana (2007a) considered the accuracy problem in the partitioning clustering of multimedia time-series. As another evidence, Gullo, Ponti, Tagarelli, Tradigo, and Veltri (2011) remarked that special characteristics of time-series data, have posed challenges to the effective identification of clusters. Hence, the problem of this study is stated briefly as:

"Applying conventional clustering algorithms to accurately and meaningfully identify clusters of time-series data is difficult because time-series data are naturally large, high dimensional, and consist of noise and outliers."

To address this problem, the reasons for the low accuracy in time-series clustering were investigated. Generally speaking, time-series clustering accuracy suffers from overlooking of data, inaccurate distance measure and inappropriate clustering algorithms. To support the problem statement, some scenarios are explained in the following according to the mentioned reasons:

1.3.1 Overlooking of Data

First, time-series datasets are very large. Hence, the dimension of time-series data should be reduced due to limited space and memory. As a result, data are overlooked, that is, the important points of the series which appear in data over time are lost. Therefore, some significant parts of time-series may not play their role in the final clusters which reflect the system (such as peaks in stock price). For example, the daily blood pressure of a patient is a time-series, the time points of which are the average of (or a sample of) the patient's blood pressure for each day for a specific period. Samples

of blood pressures are usually taken two to three times a day, but time-series is represented on a daily basis because of a reduction in its dimensions. The daily time-series of two patients can be very similar, which puts them in the same group (Lai, Chung, & Tseng, 2010). However, an in-depth look into the time-series of blood pressures may reveal that the blood pressure of a patient may vary and is more frequent than that of another patient. For example, one may lose some dangerous points of low blood pressure for a patient, as illustrated in Figure 1.1. This clearly shows why accurate clustering of time-series is important.

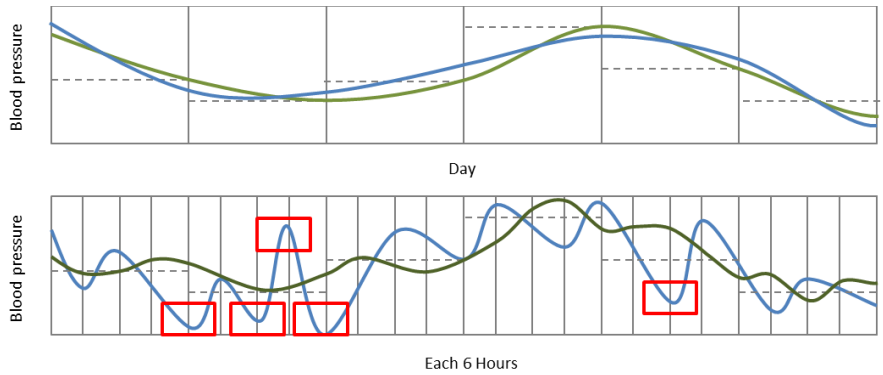


Figure 1.1: The time-series of blood pressure of two patients in two different granularities. Adopted from Lai et al. (2010)

As a result, given a time-series dataset, a clustering algorithm may generate different results when different time granularities are considered (see Appendix A for more examples). However, many sensitive systems (e.g., financial or healthcare systems) find that generating such unstable and inaccurate results is unacceptable for various representations of time-series related to a unique problem (see Section 5.3.1.1.1).

1.3.2 Inaccurate Similarity Measure

Similarity/dissimilarity measure is a core task of data mining. Similarity/dissimilarity measurement between time-series is not as simple as that for static objects because the order of points in the time-series should be taken into account. Various distance measures are given in the literature (see Section 2.5). However, unique superior measure exists because it is highly dependent on type, the characteristics of time-series,

and the representation method used. For example, a financial time-series (stock data) has its own characteristics where salient points are important but in another time-series, the trend or frequency may be more significant in similarity measurement (Lkhagva, Suzuki, & Kawagoe, 2006). Moreover, similarity/dissimilarity measure among time-series is highly dependent on the representation method used for dimensionality reduction (due to compatibility purposes). Finally, similarity measure suffers from shifting, noise and outliers of time-series in a high-dimensional space (see Section 2.5). For example, using Euclidean distance on time-series may generate wrong clusters (Keogh & Ratanamahatana, 2004) because it disregards the shifts, while another accurate distance measure such as Dynamic Time Warping (DTW) may solve the problem by handling the shifts, as depicted in Figure 1.2. However, DTW and most state-of-the-art similarity matching methods are quadratic in time and do not work in reality, especially on large datasets (Salvador & Chan, 2007). Moreover, DTW may not generally work on all datasets, as explained in Section 6.2.3. As a result, providing an accurate and fast distance measure is not a trivial task, and using imprecise measures may result in incorrect clusters. This issue is a major drawback in many sensitive systems.

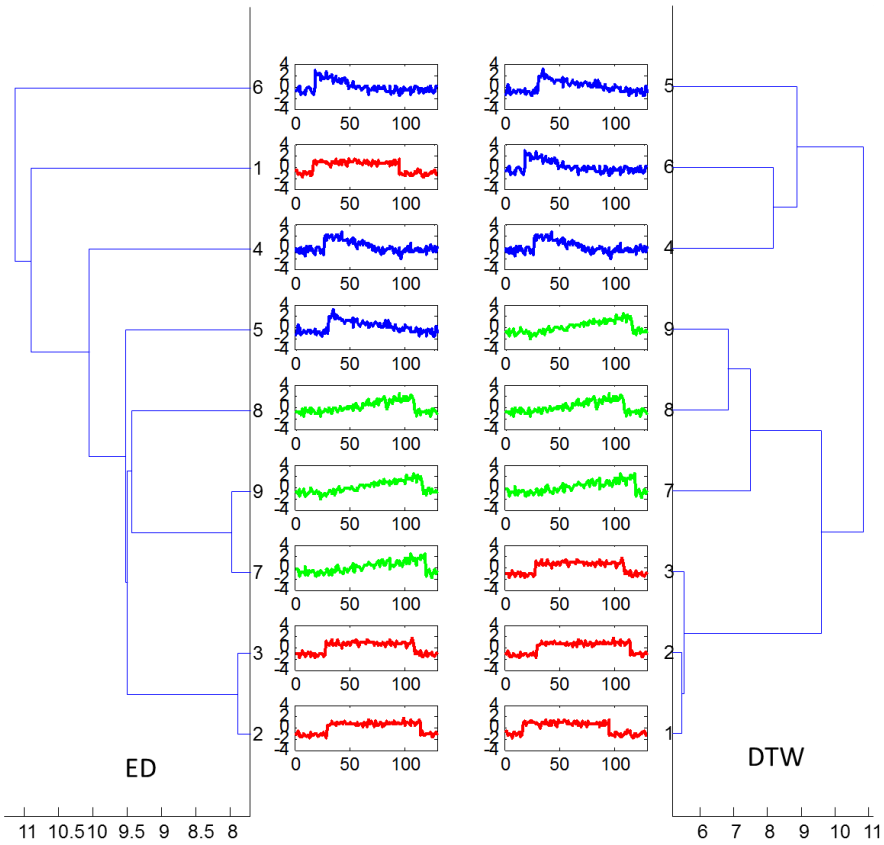


Figure 1.2: Clustering examples of CBF dataset using DTW and ED

1.3.3 Inappropriate Algorithms

Static data is derived from an object with values that do not change or change insignificantly over time. Most, if not all, developed clustering algorithms are compatible with static data. On the other hand, unlike static data, time-series data change dynamically. Previous studies (as discussed in Sections 2.4 and 2.5) imply that most systems make use of conventional algorithms (originally designed for static data) for time-series clustering. However, because time-series are high dimensional and clustering them is computationally expensive, a representation method or an adaptive distance measure is utilized to customize the conventional algorithms (Warrenliao, 2005). This leads to inaccuracy of final results due to overlooking of data (by representation) or incorrect similarity matching (by inaccurate distance calculation).

To sum up, in many systems, using conventional clustering algorithms is not practical for time-series data and thus, the absence of an adequate clustering model for time-

series is apparent. Although a trade-off exists between the accuracy and speed of clustering of time-series (which corresponds to representation method and distance measure), researchers argue that an appropriate model can resolve this issue. In this thesis, a novel model for clustering of large time-series datasets is presented to remedy the inaccuracy of results. In relation to large datasets, the first things that come to mind are speed and time execution but the problem here is NOT to perform clustering fast but rather to do so in an accurate and meaningful way for sensitive systems.

1.4 Research Questions

The research questions which are answered through this study are as follows:

Q1. Is there any alternative approach for increasing the accuracy of clustering of symbolised time-series?

Q2. Which clustering algorithm is more suitable for dimensionality reduced time-series?

Q3. How can constructed clusters be revised as post-clustering action to achieve better results?

Q4. Is there any alternative approach to run the proposed model as an interactive clustering approach?

Q5. How much does the proposed model improve the accuracy on different datasets?

1.5 Research Objectives

The main goal in this research is to improve the accuracy of time-series clustering, to produce more intuitive and meaningful clusters. The main objectives are threefold:

1. To propose and develop a new clustering model that accepts large raw time-series data as input, and generates accurate clusters without violating the time execution (answering to the first three research questions).
2. To extend the proposed model, enabling it to run interactively.
3. To evaluate the capability of the proposed methods in improving the accuracy.

1.6 Scope of Research

To ensure that this research can achieve its set of objectives within the stipulated timeframe, the following scope of the research needs to be defined:

1. The clustering of a set of individual and discrete time-series is performed as "whole time-series clustering" (see Section 2.2.2). The focus of this study is to find the clusters of time-series which are similar in shape (see Section 2.5).
2. The focus will be whole time-series clustering with a short or modest length, not on long time-series because comparing time-series that are too long is usually not very significant (J. Lin, Etter, & DeBarr, 2008). For long time-series clustering, some global measures (e.g., seasonality, periodicity, skewness, chaos) which are obtained by statistic operations, are more important (X. Wang et al., 2006).
3. The focus is on the accuracy and meaningfulness of clusters, not its speed. Meaningful clusters indicate that the clusters should capture the natural structure of data (Tan, Steinbach, & Kumar, 2006).

1.7 Chapter Organization

This thesis consists of seven chapters. Chapter 1 presents a brief background of time-series clustering and its challenges. The aim, objectives, and scope of the research are also defined. Moreover, this chapter presents the main causes of inaccuracy in time-series clustering, and highlights them by providing some example problems.

Chapter 2 reviews existing studies on time-series clustering. It covers the definition of time-series data, clustering of time-series, cluster prototypes, different distance measures, and time-series representation methods. Then, existing time-series clustering approaches are discussed, and the strengths and weaknesses of existing works are elaborated. This chapter provides a common platform from which to launch further discussions in the next chapters.

Chapter 3 describes the research methodology utilized to achieve the research objectives. It includes a brief overview of Multi-step Time-series Clustering (MTC) and Interactive Multi-step Time-series Clustering (IMTC) model as well as its main tasks and motivation for applying different methods. Moreover, the evaluation plan is explained in this chapter.

Chapter 4 explains the main part of the thesis, which is the presentation and design of the MTC and IMTC model. It focuses on the main subject of this study: accurate clustering of large time-series data. The methods which are adopted or designed to develop the models are discussed in this chapter.

Chapter 5 describes the experimental data used in this thesis. Then, the MTC is applied on existing datasets. Many aspects of this thesis are discussed in this chapter, including the methods designed to achieve the study's objectives. The process of pre-clustering, refining the clusters, and defining the representatives of each cluster are analysed here.

Most of the research questions are answered in this chapter. Finally, the parameter settings are discussed, and the final results are reported.

Chapter 6 evaluates and discusses the proposed models from different aspects. At first, the accuracy of MTC in different datasets is evaluated. Then, experimental evaluation of the MTC is shown in comparison with rival models. The performance of IMTC is experimentally determined. At last, the scalability and sensitivity of the proposed model are discussed.

Chapter 7 concludes the research and discusses how the research objectives were met. The contribution of the research outcome is then discussed, followed by suggestions on aspects of time-series clustering that may be examined for further research.

2.0 BACKGROUND AND LITERATURE REVIEW

2.1 Introduction

Clustering is a technique used to put similar data elements into homogeneous groups without advance knowledge of the group definitions (Rai & Singh, 2010). It is a useful approach to identify structure in an unlabelled dataset. Clusters are formed in the way that objects have maximum similarity with other objects within the group, and minimum similarity with objects in other groups. A special type of clustering is time-series clustering. Time-series are dynamic objects where their feature values change over time. The data in various systems - like finances, healthcare, and business – is stored as time-series. As a result, the interest of time-series clustering has increased in various areas such as medicine, biology, finance, economics, the web, etc. It is obvious that in order to cluster time-series data, it is necessary to select a suitable representation, distance measure and clustering algorithm for the data at hand. In this chapter an overview of existing time-series clustering approaches and their strengths and weaknesses are discussed. In Section 2.2, the basics of time-series clustering are presented including the applications of time-series clustering and its taxonomy. Then *whole time-series clustering* is emphasized in Section 2.3 which is the scope of this study. In this section, the published articles related to whole time-series clustering are introduced. The major methods of each component of time-series clustering are explained including: time-series representation in Section 2.4, data similarity/dissimilarity measurement in Section 2.5, and defining prototypes in Section 2.6. Performance evaluation of time-series clustering is performed using different criterion which are explained in Section 2.7. Finally, in Section 2.8, existing *whole time-series clustering* algorithms are explained based on different categories of clustering approaches including hierarchical-based, partitioning-based, model-based, density-based, and grid-based. Moreover, focusing on accuracy problem, the recent

approaches which are very similar to this study are discussed in this section under the multi-step approaches. Subsequently, the common problems in existing works are concluded in Section 2.9.

2.2 Time-series Clustering

With increasing power of data storages and processing, real-world applications have found this chance to store and keep data for a long time. As a result, data in many applications is being stored in the form of time-series data, for example sales data, stock prices (e.g., value of Google stock), exchange rates in finance, weather data (e.g., annual rainfall), biomedical measurements(e.g., blood pressure and electrocardiogram measurements), biometrics data (image data for facial recognition), particle tracking in physics, and etc. Accordingly, different works are found in variety of domains such as geology (Harms & Goddard, 2001), bioinformatics (Gusfield, 1997), biology (Bar-Joseph, Gerber, Gifford, Jaakkola, & Simon, 2002), genetics (Das, Lin, Mannila, Renganathan, & Smyth, 1998), human motion analysis (Uehara & Shimada, 2002), space exploration (Honda, Wang, Kikuchi, & Konishi, 2002; Keogh, 1997a; Oates, 1999), handwriting recognition (Vuori & Laaksonen, 2002), multimedia (Niennattrakul & Ratanamahatana, 2007a; Ratanamahatana & Niennattrakul, 2006; Ratanamahatana, 2005), telecommunications (Das, Gunopulos, & Mannila, 1997) and finance (Gavrilov, Anguelov, Indyk, & Motwani, 2000; Ge & Smyth, 2000; R. Lin & Shim, 1995).

This amount of time-series data has provided the opportunity of analysing time-series for many researchers in data mining communities in the last decade. As a result, many researches and projects relevant to analysing time-series have been performed in various areas for different purposes: subsequence matching, anomaly detection, motif discovery (J. Lin, Keogh, Lonardi, Lankford, & Nystrom, 2004), indexing, clustering, classification (Keogh & Kasetty, 2003), visualization (Haigh, Foslien, & Guralnik, 2004), segmentation (Keogh, Chu, & Hart, 2004), identifying patterns, trend analysis,

summarization (J. Lin, Keogh, Lonardi, et al., 2003), and forecasting. Moreover, there are many on-going research projects aimed to improve the existing techniques (Rakthanmanon, Campana, Batista, Zakaria, & Keogh, 2012; Zakaria, Rotschafer, Mueen, Razak, & Keogh, 2012). There are many reviews and survey on time-series analysis and its applications (H. Ding, Trajcevski, Scheuermann, Wang, & Keogh, 2008; Fu, 2010; Hirano & Tsumoto, 2005).

Among all the techniques which have been applied to analyse time-series data, clustering is one of the most frequently used techniques, due to its exploratory nature, and its application as a pre-processing step in more complex data mining algorithms (C. Ding, He, Zha, & Simon, 2002; Fayyad, Reina, & Bradley, 1998; Kalpakis, Gada, & Puttagunta, 2001). There are some comprehensive surveys and reviews that focus on comparative aspects of time-series clustering experiments (Antunes & Oliveira, 2001; Kavitha & Punithavalli, 2010; Keogh & Kasetty, 2003; Laxman & Sastry, 2006; Rani & Sikka, 2012; Warrenliao, 2005) which shows a trend of increased activity.

2.2.1 Applications of Time-series Clustering

Clustering of time-series data is mostly utilized for discovery of interesting patterns in time-series datasets (Das et al., 1998; H. Wang, Wang, Yang, & Yu, 2002). This task itself, fall into two categories: The first group is the one which is used to find patterns that frequently appears in the dataset (Chiu, Keogh, & Lonardi, 2003; Fu, Chung, Ng, & Luk, 2001). The second group are methods to discover patterns which happened in datasets surprisingly (P. K. Chan & Mahoney, 2005; Keogh, Lonardi, & Chiu, 2002; Leng, Lai, Tan, & Xu, 2009; Wei, Kumar, Lolla, & Keogh, 2005). Briefly, finding the clusters of time-series can be advantageous in different domains to answer real world problems as follows:

1- Anomaly (novelty or discord) detection: Anomaly detection are methods to discover unusual and unexpected patterns which happen in datasets surprisingly (P. K. Chan & Mahoney, 2005; Keogh et al., 2002; Leng et al., 2009; Wei et al., 2005). For example, in sensor databases, clustering of time-series which are produced by sensor readings of a mobile robot in order to discover the events (Polz, Hortnagl, & Prem, 2003).
2- Recognizing dynamic changes in time-series: detection of correlation between time-series (He et al., 2011). For example, in financial databases, it can be used to find the companies with similar stock price move.
3- Prediction and recommendation using clustering: a hybrid technique combining clustering and function approximation per cluster can help user to predict and recommend (Graves D, 2010; Ito, Hiroyasu, Miki, & Yokouchi, 2009; Pavlidis, Plagianakos, Tasoulis, & Vrahatis, 2006; Sfetsos & Siriopoulos, 2004). For example, in scientific databases, it can address problems such as finding the patterns of solar magnetic wind to predict daily pattern.
4- Pattern discovery: to discover the interesting patterns in databases. For example, in marketing database, different daily patterns of sales of a specific product in a store can be discovered.

Table 2.1depicts some applications of time-series data in different domains are stated.

Table 2.1: Samples of objectives of time-series clustering in different domains

Reference	Dataset	Objective
Košmelj & Batagelj, (1990)	Country's energy consumption	Energy Consumption pattern of 23 European Countries (commercial consumption)
Van Wijk & Van Selow (1999)	Daily power consumption	Discovering consumer power consumption patterns
Ramoni, Sebastiani, & Cohen (2000)	Robot sensor data	To form prototypical representations of the robot's experiences
Fu et al. (2001)	Stock market data	Discovery patterns from stock time-series
Golay et al., (1998); Wismüller et al., (2002)	Functional MRI (fMRI)	To detect brain activity
Tran & Wagner (2002)	Speech time-series	Speaker verification
M. Kumar & Patel (2002)	Sales data from several departments of a major retail chain	To find seasonality patterns (Retail pattern)
Steinbach, Tan, Kumar, Klooster, & Potter, (2003)	Climate time-series	Discovery of climate indices
Möller-Levet, Klawonn, Cho, & Wolkenhauer, (2003)	Gene expression	Identification of functionally related genes
Bagnall, Janacek, De la Iglesia, & Zhang, (2003)	Time-series representing the per capita personal income	Personal income pattern
Shumway,(2003)	Earthquake	Analysing potential violations of a Comprehensive Test Ban Treaty (CTBT)
Guan & Jiang, (2007)	Financial data	To create efficient portfolio (a group of stocks owned by a particular person or company)
(C. Guo, Jia, & Zhang, 2008)	Stock exchange data	Discovery patterns from stock time-series
Rebbapragada, Protopapas, Brodley, & Alcock, (2009)	Astronomical data (star light curves)	Pre-processing for outlier detection
Gullo et al., (2011)	Mass spectra data	Exploring, identifying, and discriminating pathological cases from MS clinical samples
Kurbalija et al., (2012)	Human behaviour data	Analysis of human behaviour in psychological domain

2.2.2 Taxonomy of Time-series Clustering

In reviewing the literature, one can conclude that most works related to clustering time-series are classified into three categories: "whole time-series clustering", "subsequence clustering" and "time point clustering" as depicted in Figure 2.1. The first two categories are mentioned in (2005). "**Whole time-series clustering**" is considered as clustering of a set of individual time-series with respect to their similarity. Here, clustering means applying conventional (usually) clustering on discrete objects, where objects are time-series. "**Subsequence clustering**" means clustering on a set of subsequences of a time-series that are extracted via a sliding window, that is, clustering of segments from a single long time-series. Additionally, there is another category of clustering which is "**Time point clustering**" seen in some papers (Gionis & Mannila, 2003; Morchen, Ultsch, Mörchen, & Hoos, 2005; Ultsch & Mörchen, 2005). It is clustering of time points based on a combination of their temporal proximity of time points and the similarity of the corresponding values. This approach is similar to time-series segmentation. However, it is different from segmentation from this sense that all points do not need to be assigned to clusters, i.e., some of them are considered as noise.

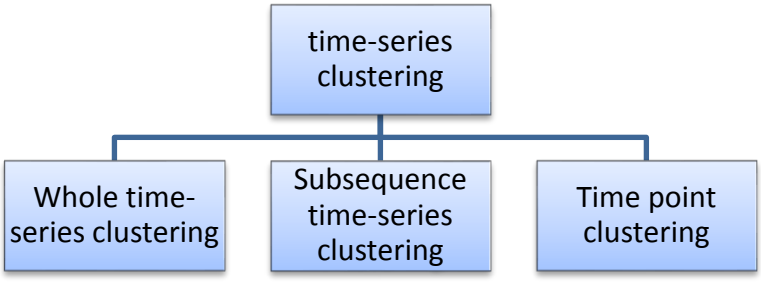


Figure 2.1: Time-series clustering taxonomy

Essentially sub-sequence clustering is performed on a single time-series, and is shown that is meaningless (Keogh & Lin, 2005). Time-point clustering also is applied on a single time-series, and is similar to time-series segmentation. That is, the objective of time-point clustering is finding the clusters of time-point instead of clusters of time-

series data. Hence, in this thesis the emphasis is on the whole time-series clustering. In the next section, whole time-series clustering and its challenges are explained.

2.3 Whole Time-series Clustering

A complete review on whole time-series clustering is performed and shown in Table 2.4. In reviewing the literature, various techniques have been recommended for the clustering of whole time-series data. However, most of them take one of the following approaches to cluster time-series data:

1) Customizing the existing conventional clustering algorithms (which work with static data) such that they become compatible with the nature of time-series data. In this approach, usually their distance measure (in conventional algorithms) is modified to be compatible with the raw time-series data (T. W. Liao & Warrenliao, 2005).

2) Converting time-series data into simple objects (static data) as input of conventional clustering algorithms (T. W. Liao & Warrenliao, 2005).

3) Using multi resolutions of time-series as input of a multi-step approaches. This approach is discussed further in Section 2.8.6.

Beside this common characteristic, there are generally three different ways to cluster time-series, namely shape-based, feature-based and model-based. Figure 2.2 shows a brief of these approaches. In the **shape-based** approach, the shapes of two time-series are matched as well as possible, by a non-linear stretching and contracting of the time axes. This approach has also been labelled as a raw-data-based approach because it typically works directly with the raw time-series data. Shape-based algorithms usually employ conventional clustering methods which are compatible with static data while their distance/similarity measure has been modified with an appropriate one for time-series. In the **feature-based** approach, the raw time-series are converted into a feature vector of lower dimension. Later, a conventional clustering algorithm is applied to the

extracted feature vectors. Usually in this approach, an equal length feature vector is calculated from each time-series followed by the Euclidean distance measurement (Hautamaki, Nykanen, & Franti, 2008). In **model-based** methods, a raw time-series is transformed into model parameters (a parametric model for each time-series,) and then a suitable model distance and a clustering algorithm (usually conventional clustering algorithms) is chosen and applied to the extracted model parameters (Warrenliao, 2005). However, it is shown that usually model-based approaches has scalability problems (Vlachos, Gunopulos, & Das, 2004), and its performance reduces when the clusters are close to each other (Mitsa, 2009).

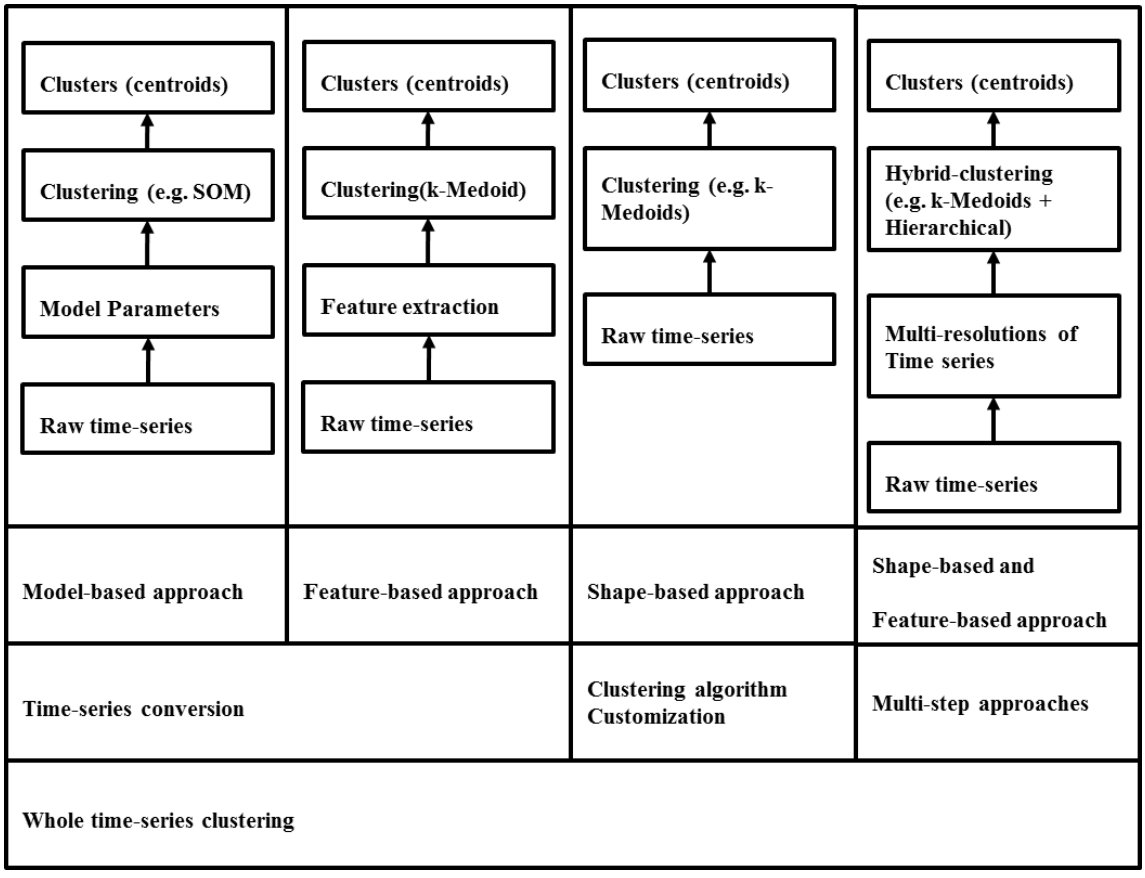


Figure 2.2: The time-series clustering approaches

As explained in Section 1.3, "whole time-series clustering" approaches are suffering from overlooking data (by feature-based approach) and high complexity of distance measures (by shape-based approaches). The approach proposed in this thesis is using a multi-step approach with composition of shape-based and feature-based techniques to

use their strengths and overcome their drawbacks. That is, the proposed model in this research uses time-series data in low-resolution which reduce the complexity of clustering. Moreover, data are used in high-resolution mode to generate the prototypes and to avoid overlooking of data. In the proposed model, the transferred time-series is used for pre-clustering and high-dimensional time-series for revising the results. The motivation for using a multi-step approach is discussed further in Section 3.2.4.

Reviewing existing works in the literature, it is implied that essentially time-series clustering has five components: dimensionality reduction (representation) method, distance measurement, clustering algorithm, prototype definition, and evaluation. Figure 2.3 shows an overview of these components.

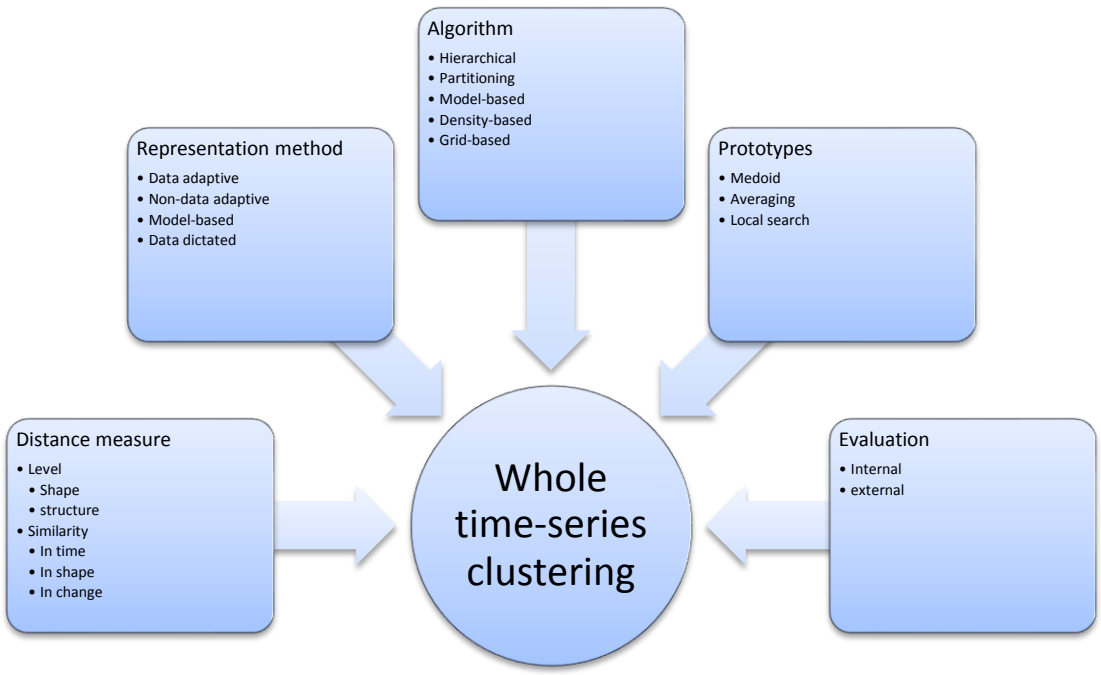


Figure 2.3: An overview of five components of whole time-series clustering

The general process in the time-series clustering uses some/all of these components depending on the problem. Usually, data is approximated (using a representation method) in such a way that can fit in memory. Then, a clustering algorithm is applied on

data using a distance measure. In the clustering process, usually a prototype is required for summarization of the time-series. At last, the clusters are evaluated using criteria.

In the following sub-sections, each component is discussed, and several related works and methods are reviewed.

2.4 Time-series Representation

The first component of time-series clustering explained here is dimension reduction. Applying a dimension reduction is a common solution for most whole time-series clustering approaches proposed in the literature (G. Duan, Suzuki, & Kawagoe, 2006; Ghysels, Santa-Clara, & Valkanov, 2006; Keogh, 2005; J. Lin, Keogh, Lonardi, et al., 2003). This section looks at methods of time-series representation (data reduction).

Definition 2.1: Time-series representation, given a time-series data $F_i = \{f_1, .., f_t, .., f_T\}$, representation is transforming the time-series to another dimensionality reduced vector $F'_i = \{f'_1, .., f'_x\}$ where x<T and if two series are similar in the original space, then their representations also should be similar in the transformation space.

Based on Ratanamahatana et al., (2005) ,"the key to the efficiency and accuracy of the solution is to choose an appropriate data representation method". High dimensionality and noise are characteristics of most time-series data (Keogh & Kasetty, 2003). Dimensionality reduction methods are usually used in whole time-series clustering in order to address these issues and promote the performance. For example, Figure 2.4 shows a time-series related to balance of transactions of a bank customer, its reconstructed coefficients and approximation coefficients as an example of time-series representation in different levels.

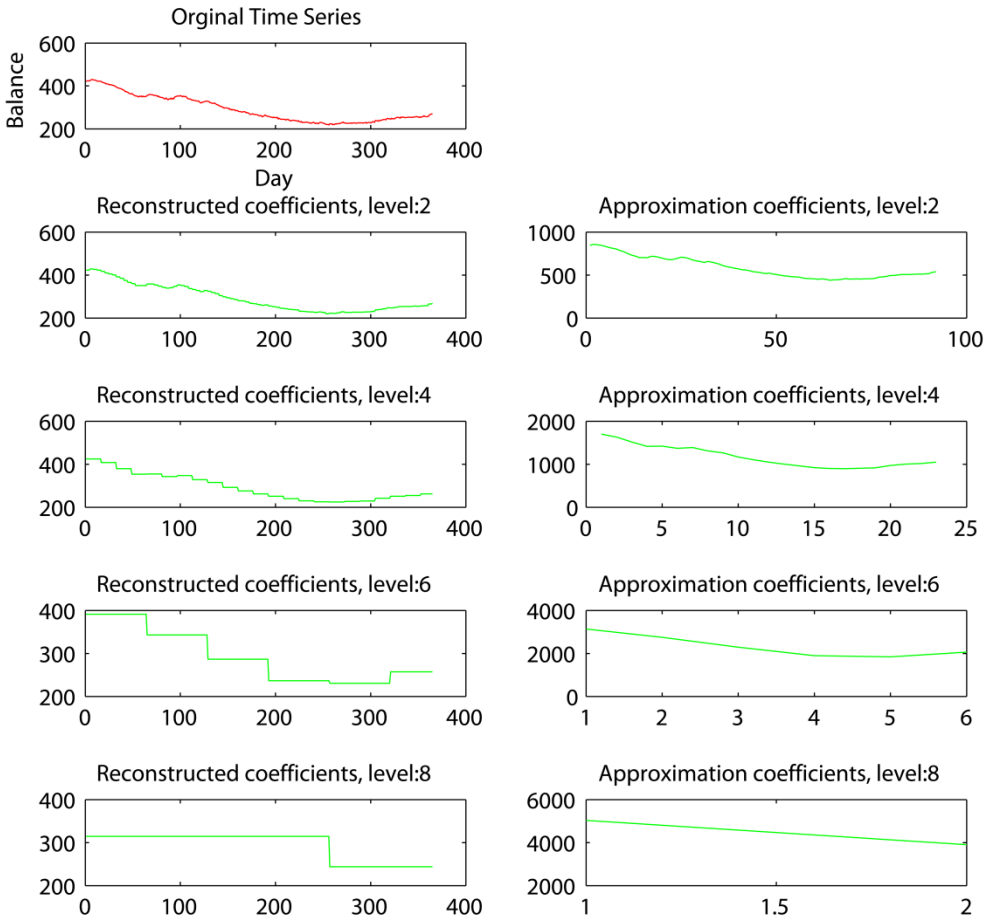


Figure 2.4: A sample of dimensionality reduction of time-series. Left: Coefficients of a time-series constructed from 'Haar' wavelet decomposition structure. Right: the approximation coefficients of the time-series.

Time-series reduction techniques have progressed a long way and are widely used with large scale time-series dataset, each with their own features and drawbacks. Accordingly, many researches had been carried out focusing on representation and dimensionality reduction (K. Chan & Fu, 1999; Keogh, Chakrabarti, Pazzani, & Mehrotra, 2001b; Keogh & Pazzani, 1998; J. Lin, Keogh, Wei, & Lonardi, 2007; Popivanov & Miller, 2002; Y. L. Wu, Agrawal, & El Abbadi, 2000; Yi & Faloutsos, 2000). It is worth here to mention about the one of the last comparisons on representation methods. H. Ding et al. (2008) have performed a comprehensive comparison of 8 representation methods on 38 datasets. Although, they had investigated the indexing effectiveness of representation methods, the results are advantageous for clustering purpose as well. They use tightness of lower bounds to compare

representation methods. They show that there is very little difference between recent representation methods.

In taxonomy of representations, there are generally four representation types (Bagnall, Ratanamahatana, Keogh, Lonardi, & Janacek, 2006; J. Lin, Keogh, Lonardi, et al., 2003; Ratanamahatana et al., 2005; Shieh & Keogh, 2009): data adaptive, non-data adaptive, model-based and data dictated representation approaches as depicted in Figure 2.5.

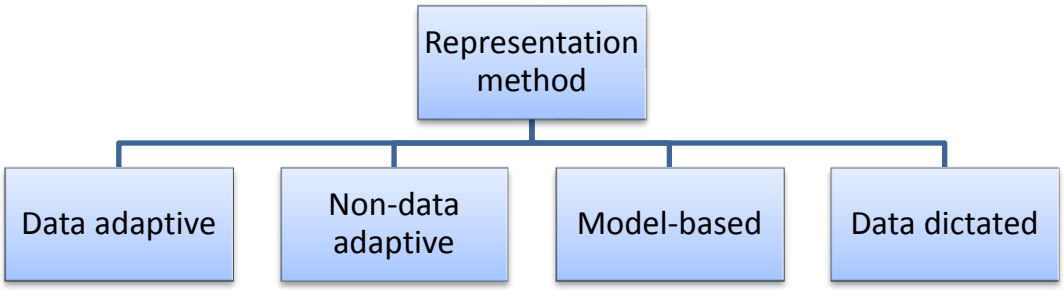


Figure 2.5: Hierarchy of different time-series representation approaches

Data adaptive representation methods are performed on all time-series in datasets and try to minimize the global reconstruction error (Xiaoyue Wang et al., 2012) using arbitrary length (non-equal) segments. This technique has been applied in different approaches such as Piecewise Polynomials Interpolation (PPI) (Morinaka, Yoshikawa, Amagasa, & Uemura, 2001), Piecewise Polynomials Regression (PPR) (Shatkay & Zdonik, 1996), Piecewise Linear Approximation (PLA), Piecewise Constant Approximation (PCA), Adaptive Piecewise Constant Approximation (APCA) (Keogh et al., 2001b), Singular Value Decomposition (SVD) (Faloutsos et al., 1994; Korn, Jagadish, & Faloutsos, 1997) , Natural Language, Symbolic Natural Language (NLG) (Portet et al., 2009), Symbolic Aggregate ApproXimation (SAX) and iSAX (J. Lin et al., 2007). Data adaptive representations can better approximate each series, but the comparison of several time-series is more difficult. **Non-data adaptive** approaches are representations which are suitable for time-series with fix size (equal-length)

segmentation, and the comparison of representations of several time-series is straightforward. The methods in this group are Wavelets (K. Chan & Fu, 1999): HAAR, DAUBECHIES, Coeiflets, Symlets, Discrete Wavelet Transform(DWT), spectral Chebyshev Polynomials (Cai & Ng, 2004), spectral DFT (Faloutsos et al., 1994), Random Mappings (Bingham, 2001), Piecewise Aggregate Approximation (PAA) (Keogh et al., 2001a) and Indexable Piecewise Linear Approximation (IPLA) (Q. Chen, Chen, Lian, & Liu, 2007). **Model based** approaches represent a time-series in a stochastic way such as Markov Models and Hidden Markov Model (HMM) (Minnen, Isbell, Essa, & Starner, 2007; Minnen, Starner, Essa, & Isbell, 2006; Panuccio, Bicego, & Murino, 2002), Statistical Models, Time-series Bitmaps (N. Kumar, Lolla, Keogh, & Lonardi, 2005), and Auto-Regressive Moving Average(ARMA) (Corduas & Piccolo, 2008; Kalpakis et al., 2001). In the data adaptive, non-data adaptive, and model based approaches user can define the compression-ratio based on the application in hand. In contrast, in **data dictated** approaches, the compression-ratio is defined automatically based on raw time-series such as Clipped (Bagnall et al., 2006; Ratanamahatana et al., 2005).

In the following table (Table 2.2) the most famous representation methods in the literature are shown.

Table 2.2: Representation methods for time-series data

Representation method	Introduced by	Used by	Complexity	Arbitrary lengths	Weighted distance	Elastic distance	Standard distance	lower bounding	Type	Usage	Pros	Cons
Chebyshev Polynomials (CHEB)	(Cai & Ng, 2004)	-	-	-	-	Not support	Support	Support	Non data adaptive, Wavelet, Orthonormal	-	-	-
Indexable Piecewise Linear Approximation (IPLA)	Q. Chen et al., (2007)	-	-	-	-	Not support	Support	-	Non data adaptive	-	-	-
Discrete Fourier Transform (DFT)	Agrawal, Faloutsos, & Swami, (1993) and Faloutsos et al., (1994)	Owsley, Atlas, & Bernard, (1997)	O(n(log(n))	Not support	Not support	Not support	Support	Support	Non data adaptive, Spectral	Natural Signals	No false dismissals.	Not support time warped queries
Discrete Wavelet Transform (DWT)	K. Chan & Fu, (1999), Kawagoe & Ueda, (2002) and Agrawal et al., (1993)	Vlachos, Lin, & Keogh, (2003) ; Shahabi, Tian, & Zhao, (2002) and H. Guo, Liu, Liang, & Gao, (2008)	O(n)	Not support	Not Support	Support	Support	Support	Non data adaptive, Wavelet	stationary signals	Better results than DFT	Not stable results, Signals must have a length $n = 2^{\text{some\_integer}}$
Discrete Cosine Transformation (DCT)	Korn et al., (1997)	-	-	-	-	Not support		Support	Non data adaptive, Spectral	-	-	-
Singular Value Decomposition (SVD)	Faloutsos et al., (1994) and Korn et al., (1997)	-	very expensive O(Mn^2)	Not support	Not support	Not support	Not support	Support	Data adaptive	text processing community	underlying structure of the data	-
Piecewise Linear Approximation (PLA)	Keogh & Pazzani, (1998)	-	O(nlogn) complexity for "bottom up" algorithm		Not support	Support	Not support	Support	Data adaptive	natural signals, biomedical	-	Not (currently) indexable, very expensive O(n^2N)
Piecewise Aggregate Approximation (PAA)	Yi & Faloutsos, (2000) and Keogh et al., (2001a)	Ge & Smyth, (2000); Perng, Wang, Zhang, & Parker, (2000)	Extremely Fast O(n)	Support	Support	Support	Support	Support	Non data adaptive	-	-	-
Adaptive Piecewise Constant Approximation (APCA)	Keogh et al., (2001b)	-	O(n)	Support	Support	Support	Support	Support	Data adaptive	-	Very efficient	complex implementation
Clipped Data	Ratanamahatana	Bagnall &	-	Support	-	Not	Support	Support	Data dictated	hardware	-	Ultra compact

	et al., (2005)	Janacek, (2005) and Bagnall et al., (2006)				support						representation
Symbolic Approximation (SAX)	Keogh, Lonardi, & Ratanamahatana (2004)	J. Lin et al., (2007)	O(*n*)	Support	Support	Support	Support	Support	Data adaptive	string processing and bioinformatics	Allows Lower bounding and Numerosity Reduction	Discretize and alphabet size
perceptually important point (PIP)	Chung, Fu, & Luk (2001)	Fink & Pratt, (2003); T. C. Fu, Chung, Luk, & Ng, (2010); T. C. Fu et al., (2001); Pratt & Fink, (2002)	-	Support	Support	N/A	Not support	Not support	Non data adaptive	Finance	-	-

2.4.1 Other representation methods

Different approaches for representation of time-series data are proposed in articles. Although most of them focus on mitigation of the executing time of process (mostly focusing on indexing process), some of them consider the quality of representation. From these group of works, in Ratanamahatana et al. (2005), the authors focus on the accuracy of representation method and suggest a bit level approximation of time-series. Each time-series is represented by a bit string, and each bit value specifies whether the data point's value is above the mean value of the time-series. This representation can be used to compute an approximate clustering of the time-series. This kind of representation (clipped representation) has capability of being compared with raw time-series, but in the other representations, all time-series in dataset must be transformed into the same representation in terms of dimensionality reduction. However, clipped series are theoretically and experimentally sufficient for clustering based on similarity in change (model based dissimilarity measurement) not clustering based on shape.

In Lkhagva et al. (2006), the authors consider the importance of accuracy in financial systems and propose the Extended Symbolic Aggregate ApproXimation (ESAX) which is a symbolic representation customized for financial time-series. Later, in another work, Liu & Shao (2009) present a similarity measure based on SAX for financial time-series. They focus on the shortage of SAX representation in lacking considering dynamic information of trends. They propose a similarity measure function, Composite-Distance-Function which joins point distance advantages and trend-distance advantages together.

The Perceptually Important Points (PIPs) posed by Chung et al. (2001) is one of the first motivations for a group of works on improving the accuracy of representation methods, especially in finance systems. Later, T. C. Fu, Chung, Luk, & Ng (2007) used the concept of PIP to propose a new representation where the important points of time-

series are collected by measuring the distance between the time-series points and their trend. In this representation, the point that has the largest distance is chosen. Then, Bao (2007), suggests a generalized model in financial time-series using turning points in financial data. Turning points are the important points which is presented by Fu et al. (2007). Then, in another study (Phetking, Sap, & Selamat, 2008), the authors focus on the problem of overlooking important points in financial time-series after transformation into another representation. Consequently, they propose a multiresolution important point retrieval method for financial time-series representation. They use the most Important points (MIPs) which comprises of the most peak (MP) points and the most dip (MD) points in time-series. Ultimately, Fu et al. (2010) poses accuracy problem related to indexing financial time-series and present an indexing approach based on clustering. They propose a time-series indexing framework based on data point importance for dimensional reduction. Then, they use important points to increase the quality of representation.

2.4.2 Discussion

There are many works related to representation of time-series which were categorized and discussed in details in Section 2.4. However, it is undeniable that as more dimensionally reduction occurs, more data is lost and becomes inaccurate, and consequently less time execution. Finding a trade-off between the accuracy and speed is a controversial and non-trivial task in representation methods. That is, a threshold of dimensionality reduction should be found which is a subjective issue and is highly dependent on application in hand, and the type of time-series in dataset. However, among all these representation methods which have their strong points and weaknesses, in this thesis, the focus is on Symbolic Aggregate ApproXimation (SAX) representation because of its strength in representation as described in Section 2.4.4. However, at first

a review is performed about Piecewise Aggregate Approximation (PAA) representation which is the base of SAX representation.

2.4.3 Brief Review of PAA

Two different studies (Keogh et al., 2001a; Yi & Faloutsos, 2000), separately approximated the time-series using segmentation approach. They use mean value of the equal-length segmentations of time-series as approximate value of that part (dotted lines). This technique is called Piecewise Aggregate Approximation (PAA), and is quite fast (Keogh et al., 2001b), can be used for arbitrary length queries, and able to handle different distances measures (Ge & Smyth, 2000; Perng et al., 2000). Figure 2.6 shows a time-series with 32 data points which is represented by a 8-dimensional time-series using PAA. If $F_i = \{f_1, .., f_t, .., f_T\}$ is considered as a time-series, then, PAA discretized time-series to $\bar{F}$ where $\bar{F} = \bar{f_1}, \ldots, \bar{f_w}$. Each segment of $\bar{F}$, i.e., $\bar{f_i}$, is a real value which is the mean of all data points in the i[th] segment of F In the following figure, $\bar{f_8} = -0.72$.

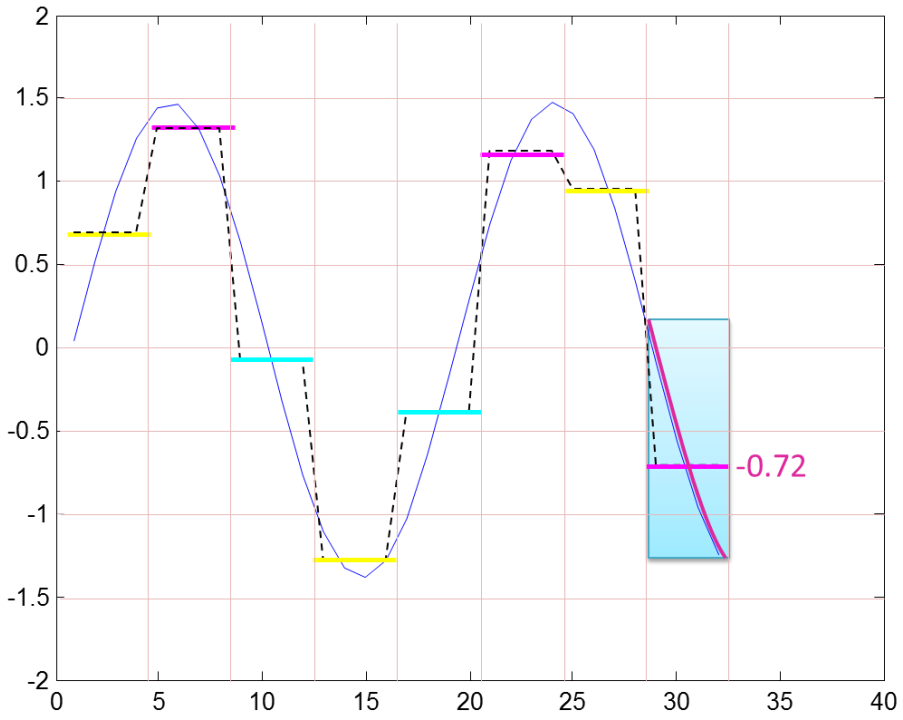


Figure 2.6: Representation of time-series by PAA

PAA is competitive with Fourier transform, Wavelet and many more sophisticated approaches in terms of quality and speed (K. Chan & Fu, 1999; Keogh et al., 2001a; Yi & Faloutsos, 2000). However, PAA has a bad quality in representing various movement shapes of time-series in lower dimension space, because of its smoothing process by averaging (Park & Lee, 2010) .

2.4.4 Brief Review of SAX

In this thesis, Symbolic Aggregate ApproXimation (SAX) transformation is adopted in order to reduce the dimension of time-series data. SAX is a symbolic representation of time-series developed by Keogh et al. in 2003 and has been used by more than 50 groups in different data mining researches (J. Lin et al., 2007). This method is a two-step process which transfers a time-series into the Piecewise Aggregate Approximation (PAA) representation as explained in Section 2.4.3 and then it maps the coefficients to symbols.

Let consider $\bar{F} = \bar{f}_1, \ldots, \bar{f}_w$ as discretized time-series by PAA transformation. Then, $\hat{F}$ where $\hat{F} = \hat{f}_1, \ldots, \hat{f}_w$ is defined by mapping the PAA coefficients to 'a' SAX symbols. 'a' is the alphabet size (e.g., for the alphabet= {a, b, c, d, e, f}, 'a' = 6), and the alphabets in SAX, are defined by "breakpoints". Based on Keogh definition, a list of numbers $B = \beta_1 \ldots \beta_{a-1}$ is defined as "Breakpoints" to determine the area of each symbol in SAX transformation. These sorted numbers, produce equal size areas under Gausian curve as illustrated in Figure 2.7. It is based on the fact that normalized time-series have highly Gaussian distribution (J. Lin et al., 2007).

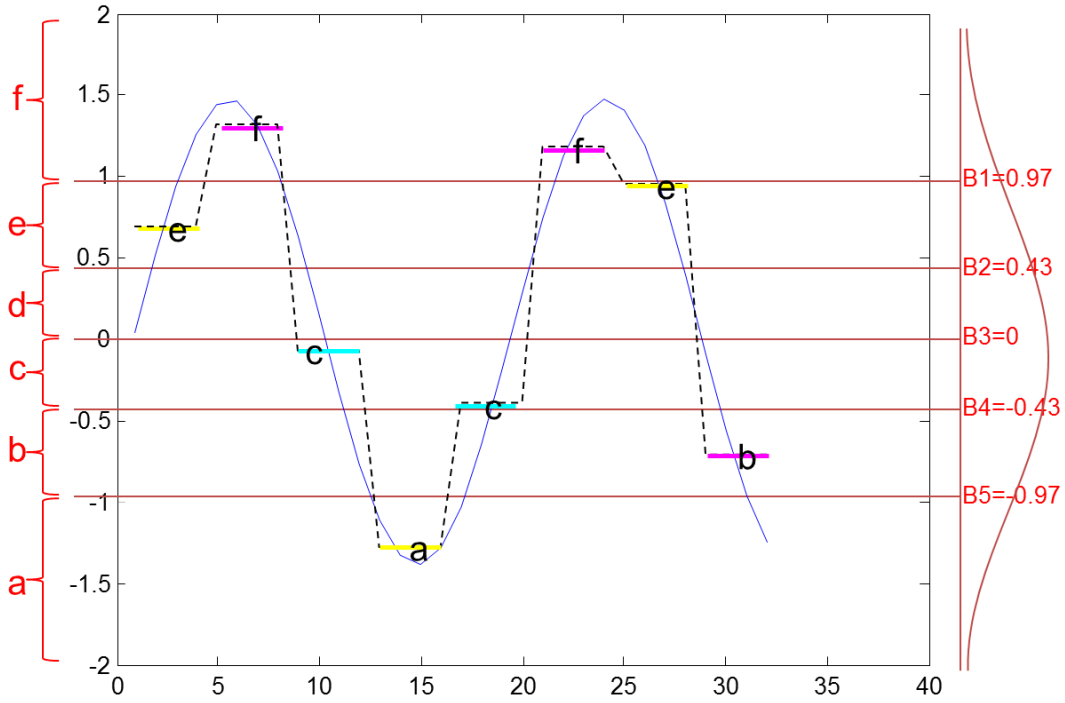


Figure 2.7: Symbolic representation of time-series by SAX based on PAA

In Figure 2.8, the raw time-series and symbolic representation of time-series are depicted. They are obtained by PAA representation of raw time-series of the behaviour of a bank's customer credit card bills over the time span of one year.

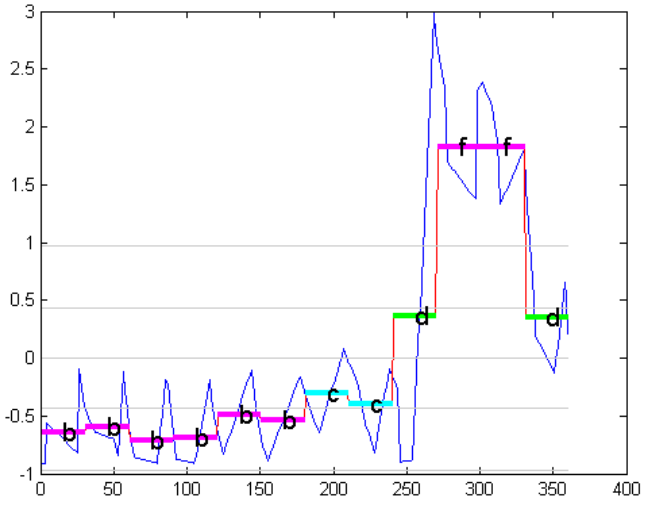


Figure 2.8: Symbolic representation of a sample raw time-series by SAX

Now the question is "Why symbolization?" Basically, a discretization technique that will produce symbols with equiprobability is desirable (Apostolico, Bock, & Lonardi, 2003; Lonardi, 2001; Mörchen & Ultsch, 2005). When time-series is transferred as a

symbolic representation, e.g., using SAX, it needs less memory which is very essential in applications which deal with long or huge amount of time-series data (J. Lin et al., 2007). In this kind of applications, fetching time-series data from disk to memory is a big challenge, and appropriate compressing of data is highly important. Considering clustering algorithms which typically require the recalculation of models directly on main memory for each iteration, symbolic representation of data will speed up the time execution of clustering algorithm. As one evidence, considering this principle, Bagnall and Janacek (2005), use clipped series (as a special case of SAX) to provide time improvement for clustering algorithms. Moreover, they mention that using symbolic representation instead of raw time-series, improves the accuracy of clustering in the presence of outliers. To sum up, SAX is as good as other well-known and often-used representation methods, such as Discrete Wavelet Transform (DWT) and Discrete Fourier Transform (DFT), while it requires less storage space (J. Lin et al., 2007).

2.5 Similarity/Dissimilarity Measure

This section looks at methods of distance measurement to choose the best similarity measure among time-series being compared. The theoretical issue of time-series similarity/dissimilarity search is proposed by Agrawal et al. (1993) and subsequently it became a basic theoretical issue in data mining community.

Time-series clustering relies on distance measure to a high extent. There are different measures which can be applied to measure the distance among time-series. Some of similarity measures are proposed based on a specific time-series representations, for example, MINDIST which is compatible with SAX (J. Lin et al., 2007), and some of them work regardless of representation methods, or are compatible with raw time-series. In traditional clustering, distance between static objects is exact match based but in time-series clustering, distance is calculated approximately. In particular, in order to compare time-series with irregular sampling intervals and length, it is of great

significance to adequately determine the similarity of time-series. There is different distance measures designed for specifying similarity between time-series. The Hausdorff distance and modified Hausdorff (MODH), HMM-based distance, Dynamic Time Warping (DTW), Euclidean distance, Euclidean distance in a PCA subspace, and Longest Common Sub-Sequence (LCSS) are the most popular distance measurement methods used for time-series data. References on distance measurement methods are given in Table 2.3.

One of the simplest ways for calculating distance between two time-series is considering time-series as univariate time-series, and then calculating the distance measurement across all time points.

Definition 2.2: Univariate time-series, a univariate time-series is the simplest form of temporal data and is a sequence of real numbers collected regularly in time, where each number represents a value (X. Wang et al., 2004).

Definition 2.3: Time-series distance, let $F_i = \{f_{i1}, .., f_{it}, .., f_{iT}\}$ be a time-series of length T. If the distance between two time-series is defined across all time points, then $\mathrm{dist}(F_i, F_j)$ is the sum of the distance between individual points

$$\mathrm{dist}(F_i, F_j) = \sum_{t=1}^{T} \mathrm{dist}(f_{it}, f_{jt}) \quad 2.1$$

In shape-based distance measuring of time-series, researches done on this domain usually have to challenge with some problems. These include noise, amplitude scaling, offset translation, longitudinal scaling, linear drift, discontinuities and temporal drift which are the common properties of time-series data. These properties are broadly investigated in the literature (Keogh & Pazzani, 1998).

The choice of a proper distance approach depends on the characteristic of time-series, its length of time-series, representation method, and of course on the objective of clustering time-series to a high extent. This is depicted in Figure 2.9.

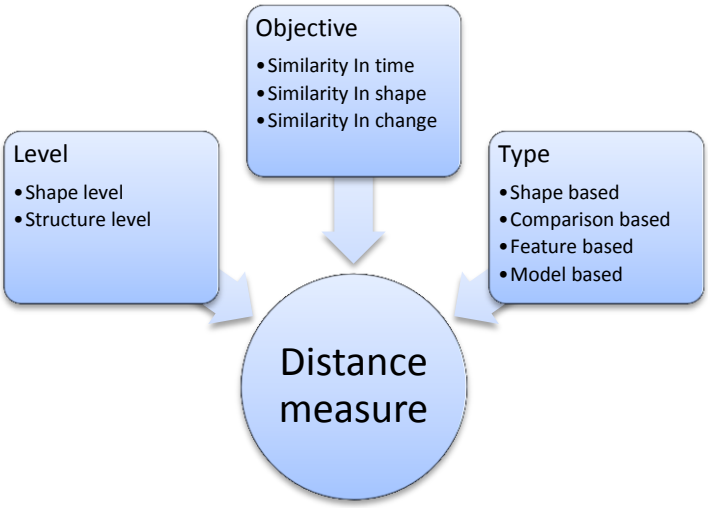


Figure 2.9: Distance measure approaches in the literature

Typically, there are three objectives which respectively require different approaches (Bagnall & Janacek, 2005):

1) Finding similar time-series in time: because this similarity is on each time step, correlation based distances or Euclidean distance measure are proper for this objective. However, because this kind of distance measuring is costly on raw time-series, the calculation is performed on transformed time-series, such as fourier transforms, wavelets or Piecewise Aggregate Approximation (PAA). Keogh and Kasetty (2003), have done an comprehensive review on this matter. Clustering of time-series that are correlated, (e.g., to cluster time-series of share price related to many companies to find which shares change together and how they are correlated) is categorized as clustering based on similarity in time (Bagnall & Janacek, 2005; Ratanamahatana et al., 2005).
2) Finding similar time-series in shape: the time of occurrence of patterns is not important to find similar time-series in shape. As a result, elastic methods

(Agrawal et al., 1993; Aref, Elfeky, & Elmagarmid, 2004) such as Dynamic time Warping (DTW) (Chu, Keogh, Hart, Pazzani, & others, 2002) is used for dissimilarity calculation. Using this definition, clusters of time-series with similar patterns of change are constructed regardless of time points, for example, to cluster share price related to different companies which have a common pattern in their stock independent on its occurrence in time-series (Bagnall & Janacek, 2005). Similarity in time is a special case of similarity in shape. A research has revealed that similarity in shape is superior to metrics based on similarity in time (Ratanamahatana & Keogh, 2005).

3) Finding similar time-series in change (structural similarity): In this approach, usually modeling methods such as Hidden Markov Models (HMM) (Smyth, 1997) or an ARMA process (Kalpakis et al., 2001; Xiong & Yeung, 2002) are utilized, and then similarity is measured on the parameters of fitted model to time-series. That is, clustering time-series with similar autocorrelation structure, e.g., clustering of shares which have a tendency to increase after a fall in share price in the next day (Bagnall & Janacek, 2005). This approach is proper for long time-series, not for modest or short time-series (X. Wang et al., 2006).

Clustering approaches could be classified into two categories based on the length of time-series: "shape level" and "structure level". The "**shape level**" is usually utilized to measure similarity in short-length time-series clustering such as expression profiles or individual heartbeats by comparing their local patterns, whereas "**structure level**" measures similarity which is based on global and high level structure, and it is used for long-length time-series data such as an hour's worth of ECGs or yearly meteorological data (X. Wang et al., 2006). Focusing on shape-based clustering of short length time-series, in this study, shape level similarity is used. Depend on the objective and length of time-series, the proper type of distance measures is determined. Essentially, there are

four types of distance measure in the literature. Please refer to Table 2.3 for references on the types of distance measure. *Shape-based similarity* measure is to find the similar time-series in time and shape, such as Euclidean, DTW (Sakoe & Chiba, 1971, 1978), LCSS (Banerjee & Ghosh, 2001; Vlachos, Kollios, & Gunopulos, 2002), MVM (Latecki et al., 2005). It is a group of methods which are proper for short time-series. *Compression based similarity* is suitable for short and long time-series, such as CDM (Keogh et al., 2007), Autocorrelation, Short time-series distance (Möller-Levet et al., 2003), Pearson's correlation coefficient and related distances (Rodgers & Nicewander, 1988), Cepstrum (Kalpakis et al., 2001), Piecewise normalization (Indyk & Koudas, 2000) and Cosine wavelets (Huhtala & Karkkainen, 1999). *Feature based similarity* measure are proper for long time-series, such as Statistics, Coefficients, *Model based similarity* is proper for long time-series, such as HMM (Smyth, 1997) and ARMA (Kalpakis et al., 2001; Xiong & Yeung, 2002).

A survey on various methods for efficient retrieval of similar time-series were given by Last and Kandel (2004). Furthermore, in (Warrenliao, 2005), authors have presented the formulas of various measures. Then, Zhang et al. (2006) have performed a complete survey on the aforementioned distance measurements and compared them in different applications. In Table 2.3, different measures are compared in terms of complexity and their characteristics.

Table 2.3: Similarity measure approaches in the literature

Distance measure	Defined by	Used in	Complexity	Length support	Method	Similarity type	Noise robustness	Characteristics
Euclidean distance (ED)	Faloutsos et al.(1994)	(Golay et al., 1998; Keogh & Kasetty, 2003; Keogh, Wei, Xi, Lee, & Vlachos, 2006)	O(n)	short time-series	Shape-based	similarity in time	No	Lock-step Measure (one-to-one) using in indexing, clustering and classification, Sensitive to scaling.
Dynamic Time Warping (DTW)	Sakoe & Chiba (1971, 1978)	(Berndt & Clifford, 1994; Chu et al., 2002; Hu, Ray, & Han, 2006; Keogh & Kasetty, 2003; Keogh & Ratanamahatana, 2004; Sankoff & Kruskal, 1983; Yu, Dong, Chen, Jiang, & Zeng, 2007)	$O(n^2)$, $O(\delta n)$ by restricting the warping path	short time-series	Shape-based	similarity in shape	No	Elastic Measure (one-to-many/one-to-none) Very well in deal with temporal drift. Better accuracy than Euclidean distance (Aach & Church, 2001) , (Chu et al., 2002) ,(Vlachos et al., 2002) , (Yi & Faloutsos, 2000) . Lowe efficiency than Euclidean distance and triangle similarity.
Longest Common Sub-Sequence (LCSS)	Banerjee & Ghosh, (2001); Vlachos et al., (2002)	(Aghabozorgi, Saybani, & Wah, 2012)	$O(n*\delta)$	short time-series	Shape-based	similarity in shape	Yes	Noise robustness
Minimal Variance Matching (MVM)	Latecki et al., (2005)		$O(mn^2)$	short time-series	Shape-based	N/A	-	Automatically skips outliers
Short time-series distance (STS)	Möller-Levet, Klawonn, Cho, & Wolkenhauer (2003)		-	short and long time-series	Feature-based	N/A	-	Sensitive to scaling. can capture temporal information, regardless of the absolute values

probability-based distance	Kumar & Patel (2002)		-	long time-series	Compression based dissimilarity	N/A	-	Able to cluster seasonality patterns
KL distance	Dahlhaus (1996)		-	long time-series	Compression based dissimilarity	N/A	-	-
J divergence	Shumway (2003)	(Shumway, 2003)	-	short time-series	Shape-based	similarity in time	-	-
Triangle similarity measure	Zhang, Wu, Yang, Ou, & Lv (2009)	(X. Zhang, Liu, Du, & Lv, 2011)	-	short time-series	Shape-based	similarity in time	-	can deal with noise, amplitude scaling very well and deal with offset translation, linear drift well in some situations (X. Zhang et al., 2009).
cross-correlation based distances	Golay et al.(1998)	(Goutte et al., 1999)	-	short time-series	Shape-based	similarity in shape	-	noise reduction, able to summarize the temporal structure
Edit Distance with Real Penalty (ERP)	L. Chen & Ng (2004)		-	short time-series	Shape-based	similarity in shape	-	Robust to noise, shifts and scaling of data, a constant reference point is used
Edit Distance on Real sequence (EDR)	L. Chen, Özsu, & Oria (2005)		-	short time-series	Shape-based	similarity in shape	-	Elastic measure (one-to-many/one-to-none), uses a threshold pattern
DISSIM	Frentzos, Gratsias, & Theodoridis (2007)		-	short time-series	Shape-based	similarity in shape	-	Proper for different sampling rates
Sequence Weighted Alignment model (Swale)	Morse & Patel (2007)		-	short time-series	Shape-based	similarity in shape	-	Similarity score based on both match rewards and mismatch penalties.
Spatial Assembling Distance (SpADe)	Y. Chen, Nascimento, Ooi, & Tung (2007)		-	long time-series	Model based	similarity in change	-	Pattern-based Measure
Threshold Queries (TQuEST)	Aßfalg et al.(2006)	(Aßfalg et al., 2008)	-	long time-series	Model based	similarity in shape	-	Threshold-based Measure, considers intervals, during which the time-series exceeds a certain threshold for comparing time-series rather than

								using the exact time-series values.
histogram-based	L. Chen & Özsu (2005)	(J. Lin & Li, 2009)	-	short time-series	Shape-based	similarity in shape	-	Using multi-scale time-series histograms
dictionary-based compression	Lang, Morse, & Patel (2010)	-	-	long time-series	Compression based dissimilarity	N/A	-	Lang et al. (Lang et al., 2010) develop a dictionary compression score for similarity measure. A dictionary-based compression technique is suggested to compute long time-series similarity.
Compression-based dissimilarity measure(CDM)	Keogh et al.(2007)	-	-	short and long time-series	Compression based dissimilarity	N/A	-	in (Keogh et al., 2007) Keogh et al. a parameter-light distance measure method based on Kolmogorov complexity theory is suggested. Compression-based dissimilarity measure (CDM) is adopted in this paper.
Autocorrelation	C. Wang & Sean Wang (2000)	(X. Wang et al., 2004) (Keogh & Kasetty, 2003)	-	short and long time-series	Compression based dissimilarity	N/A	-	-
Pearson's correlation coefficient and related distances	Rodgers & Nicewander (1988)	(Tseng & Kao, 2007) (Harrison, 2005; Möller-Levet et al., 2003)	-	short and long time-series	Compression based dissimilarity	N/A	-	invariant to scale and location of the data points
Cepstrum	Kalpakis, Gada, & Puttagunta (2001)	(Keogh & Kasetty, 2003)	-	short and long time-series	Compression based dissimilarity	N/A	-	A spectral measure which is the inverse Fourier transform of the short-time logarithmic amplitude spectrum
Piecewise normalization	Indyk & Koudas (2000)	-	-	short and long time-series	Compression based dissimilarity	N/A	-	It involves time intervals, or "windows," of varying size. But it is not clear how to determine these "windows."
Cosine wavelets	Huhtala & Karkkainen (1999)	-	-	short and long time-series	Compression based dissimilarity	N/A	-	-
Piecewise	Keogh	(Keogh &	-	short	Compression	N/A	-	-

probabilistic	(1997)	Kasetty, 2003)		and long time-series	based dissimilarity			
Hidden Markov models (HMM)	Smyth (1997)	(J. Duan, Wang, Liu, & Xue, 2005; Horenko, 2010; Yin & Yang, 2005)	-	long time-series	Model based	similarity in change	-	able to capture not only the dependencies between variables, but also the serial correlation in the measurements
ARMA	Kalpakis et al., (2001); Xiong & Yeung, (2002)	(Kalpakis et al., 2001; Xiong & Yeung, 2004)	-	long time-series	Model based	similarity in change	-	-

2.5.1 Discussion

Choosing distance measure so that it is adequately accurate, is controversial in time-series clustering domain. There are many distance measure proposed by researchers which were compared and discussed in Section 2.5. However, the following conclusion can be drawn from literature.

1) Investigating the mentioned approaches as similarity/dissimilarity measure, it is implied that the most effective and accurate approaches are the ones which are based on dynamic programming (DP) which are very expensive in time execution (the cost of comparing two time-series is quadratic in the length of the time-series) (Salvador & Chan, 2007). Although, usually some constraints are taken for these distance/similarity measurements to mitigate the complexity (Itakura, 1975; Sakoe & Chiba, 1978), it needs careful tuning of parameters to be efficient and effective. As a result, again, a trade-off between speed and accuracy should be found in usage of this metrics. In another view, it is worthwhile to understand the extent that distance measure is effective in large scale datasets of time-series. This matter is not obtained from literature because most of the considered works are based on rather small datasets.
2) In the similarity measure researches, varieties of challenges are considered pertaining to distance measurement. A big challenge is the issue of incompatibility of distance metric with the representation method. For example, one of the common approaches that is applied to time-series analysis is based upon frequency-domain (K. Chan & Fu, 1999; Kawagoe & Ueda, 2002), while using this space, it is difficult to find the similarity among sequences and produce value-based differences to be used in clustering.
3) Euclidean distance and DTW are the most common methods for similarity measure in time-series clustering. A research has shown that, in terms of time-

series classification accuracy, the Euclidean distance is surprisingly competitive (Lkhagva et al., 2006), however, DTW also has its strength in similarity measurements which cannot be declined. Here, the focus in this thesis is on both superior approaches namely Euclidean distance and Dynamic Time Warping. The motivation for choosing these approaches, more details about these measures, and their strength and weakness are explained in Section 2.5.2 and 2.5.3.

2.5.2 Euclidian Distance (ED)

Euclidian distance is a one-to-one matching measurement which is used in most of works (about 80%) in the literature (F. K. P. Chan, Fu, & Yu, 2003; Faloutsos et al., 1994; Keogh et al., 2001a, 2001b; Keogh & Kasetty, 2003; Keogh, 1997b; Reinert, Schbath, & Waterman, 2000).

Let F_i and F_j be two time-series of length n. The Euclidian distance between F_i and F_j is defined mathematically as:

$$dis_{ED}(F_i, F_j) = \sqrt{\sum_{i=1}^{n} (f_i - f_j)^2} \qquad 2.2$$

where, the square root step can be removed because the square root function is monotonic and returns the same rankings in clustering, and classifications (Keogh & Kasetty, 2003).

Euclidian distance is simple, fast and used as benchmark in many works, because it is parameter free. However, Euclidean distance is not always the best choice as distance function. It is extremely dependent on the domain of problem in hand and characteristics of its time-series as explained in Section 5.3.3.1. Generally, there are some disadvantages in using Euclidian distance as:

1. It requires that the time-series being compared are of exactly the same dimensionality (length).
2. This measure is very weak and sensitive to small shifts across the time axis (Keogh & Ratanamahatana, 2004; Ratanamahatana & Keogh, 2005). For example it is not accurate enough for calculation similarity of sequences such as : <abaa>,<aaba>

2.5.3 Dynamic Time Warping (DTW)

In contrast to Euclidean distance which proposes a one-to-one matching, Dynamic Time Warping (DTW) is suggested as a one-to-many metric. DTW is a generalization of Euclidian distance and solves the local shift problem (out of phase points in the time axis) in the time-series to be compared. Local shift problem is a time scale issue which is a characteristic of most time-series and one-to-one matching algorithms (e.g., Euclidean distance) are not capable to handle (Berndt & Clifford, 1994; Keogh & Ratanamahatana, 2004; Ratanamahatana & Keogh, 2005; Xi, Keogh, Shelton, Wei, & Ratanamahatana, 2006). DTW is an effective approach which uses "warping" the time axis in order to achieve the best alignment between the data points within the series. In DTW each point of the sequence is matched to each element of the target time-series. That is, no elements may be skipped in a sequence.

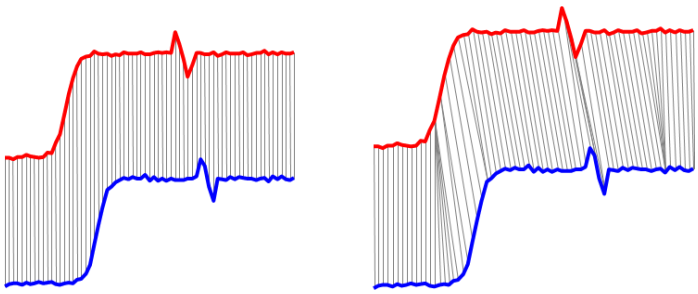


Figure 2.10: The difference between Euclidian and DTW distance (Keogh, 2006). Left: Distance calculation by ED, Right: distance calculation by DTW

Given two time-series, $F_i = \{f_1, f_2, \dots, f_i\}$, $F_j = \{f_1, f_2, \dots, f_j\}$, a $i*j$ matrix is defined where the (s, k) element of the matrix is the Euclidean distance dis(f_s, f_k) between two time points f_s and f_k. The warping path w_u from as set of warping paths =

$\{w_1, w_2, \dots, w_u\}$, that has the minimum distance between the two series of F_i and F_j is of interest.

$$dis_{DTW}(F_i, F_j) = \min(\sum_{u=1}^{U} w_u/U) \quad 2.3$$

Generally, dynamic programming is used in order to find the path effectively. In brief it can be concluded that DTW has some advantages and disadvantages as follows:

Advantages:

1. Capability of handling local shifts (temporal drift): It allows similar shapes to be matched even if they are out of phase in the time axis. It makes DTW superior to Euclidean distance (Aach & Church, 2001; Chu et al., 2002; Vlachos et al., 2002; Yi & Faloutsos, 2000).
2. DTW can assist with clustering of different-length time-series if there is no missing data. However, same-length time-series can be generated by normalization of different-length time-series without impact on accuracy (Ratanamahatana & Keogh, 2005).
3. Error rate of Euclidean distance is an order of magnitude higher than DTW (Keogh & Ratanamahatana, 2004). Moreover, a recent research (H. Ding et al., 2008), shows that DWT is more accurate than Euclidean distance.
4. The DTW is generally shown to be more robust than ED (L. Chen et al., 2005; Keogh & Ratanamahatana, 2004; Vlachos et al., 2002) and works better than edit-based measures such as LCSS, EDR, and ERP (Mitsa, 2009).

Disadvantages:

1. Sensitivity: DTW is sensitive to outliers and it can distort distance (because all points have to be matched).

2. Scalability: Speed problem, is a big challenge for DTW because it requires quadratic computation (Salvador & Chan, 2007). As a result, many researchers, (H. Ding et al., 2008; Keogh & Ratanamahatana, 2004; Kim, Park, & Chu, 2001; Xiaoyue Wang et al., 2012; Yi, Jagadish, & Faloutsos, 1998), try to speed it up usually by proposing an efficient lower bound approximations of the DTW distance to reduce its complexity. Their claim is based on this fact that DTW can be calculated very fast even for large datasets but it should be noticed that it is true under the classification problem (the search area are pruned using a lower bound distance of DTW) not under clustering problem where the distance between all objects should be calculated. However, with all progresses in speeding up the DTW (Salvador & Chan, 2007), it is still expensive (Berndt & Clifford, 1994; Xi et al., 2006) and hard to find better solution (H. Ding et al., 2008; Xi et al., 2006) specially for clustering purpose.

In this study, Euclidean distance is used for finding the similar time-series in time, and DTW for finding similar time-series in shape. In the following section, the prototyping of clusters is explained.

2.6 Cluster Prototypes

Finding the cluster prototype (representative) is an essential subroutine in time-series clustering approaches (Bagnall & Janacek, 2005; Chu et al., 2002; Corradini, 2001; Keogh & Pazzani, 1998; Rabiner & Levinson, 1979; Ratanamahatana, 2005). One of the approaches to address the low quality problem in time-series clustering is remedying the issue of inaccurate prototypes of clusters, especially in partitioning clustering algorithms such as k-Means, k-Medoids, Fuzzy C-Means (FCM), or even Ascendant Hierarchical Clustering which requires a prototype. In these algorithms, the quality of clusters is highly dependent on quality of prototypes. Given time-series in a cluster, it is clear that the cluster's prototype R_j minimizes the distance between all time-series in

the cluster and its prototype. Time-series R_j that minimizes $E(C_i, R_j)$ is called a Steiner sequence (Gusfield, 1997).

$$E(C_i, R_j) = \frac{1}{n}\sum_{x=1}^{n} dist(F_x, R_j) \ , \ \ C_i = \{F_1, F_2, .., \mathrm{F_n}\} \qquad 2.4$$

There are a few methods for calculating prototypes published in the literature of time-series, however most of these publications have not proved the correctness of their methods (Niennattrakul & Ratanamahatana, 2007b). But, in general, three approaches can be seen for defining the prototypes:

1. The medoid sequence of the set
2. The average sequence of the set
3. The local search prototype

2.6.1 Using Medoid as Prototype

In time-series clustering, most common way to approach optimal Steiner sequence is to use cluster medoid as the prototype (Kaufman, Rousseeuw, & Corporation, 1990). In this approach, the centre of a cluster is defined as a sequence which minimizes the sum of squared distances to other objects within the cluster. Given time-series in a cluster, the distance of all time-series pairs within the cluster is calculated using a distance measure such as Euclidean or DTW. Then, one of the time-series in the cluster, which has lower sum of square error is defined as medoid of the cluster (Vuori & Laaksonen, 2002). Moreover, if the distance is a non-elastic approach such as Euclidean, or if the centroid of the cluster can be calculated, it can be said that medoid is the nearest time-series to centroid.

Cluster medoid is very common among works related to time-series clustering and has been used in many papers such as (Hautamaki et al., 2008; Kaufman et al., 1990; Liao & Ting, 2006; Liao et al., 2002).

2.6.2 Using Averaging Prototype

If the time-series are in equal length, and distance metric is a none-elastic distance metric (e.g., Euclidean distance) in clustering process. Then, the averaging method is a simple averaging technique which is equal to mean of the time-series at each point. However, in the case that there are time-series with different length (Niennattrakul & Ratanamahatana, 2007b) or in the case which the similarity between time-series is based on "similarity in shape", its one-to-one mapping nature, makes it unable to capture the actual average shape. For example, in the cases that Dynamic Time Warping (DTW) or Longest Common Sub-Sequence (LCSS) are very appropriate (Gupta, Molfese, Tammana, & Simos, 1996), averaging prototype is evaded because it is not a trivial task. For more evidence, one can see many works in the literature (Bagnall & Janacek, 2005; Caiani et al., 1998; Chu et al., 2002; Corradini, 2001; Keogh & Pazzani, 1998; Oates, Firoiu, & Cohen, 2001), which avoid using elastic approaches (e.g., DTW and LCSS) where there is a need to use a prototype without providing adequate reasons (whether the clustering is based on similarity in time or shape). In the follows, two averaging methods (using DTW and LCSS) are briefly explained.

Shape averaging using Dynamic Time Warping (DTW): In this approach, one method to define the prototype of a cluster is by combination of pairs of time-series hierarchically or sequentially, for example, shape averaging using Dynamic Time Warping, until only one time-series is left (Gupta et al., 1996). The drawback about this method is its dependency on the ordering of choosing pairs which results in different final prototypes (Niennattrakul & Ratanamahatana, 2007a). Another method is the approach mentioned by Abdulla and Chow (2003), where authors proposed a cross-words reference template (CWRT), where at first, the medoid is find as the initial guess, then all sequences are aligned by DTW to the medoid, and then the average time-series is computed. The resulting time-series has the same length as the medoid, but the

method is invariant to the order of processing sequences (Hautamaki et al., 2008). In another study, the authors present a global averaging method for defining the prototypes (Petitjean, Ketterlin, & Gançarski, 2011). They use an averaging approach where the distance method for clustering or classification is DTW. However, its accuracy is dependent on the length of the initial average sequence and value of its coordinates.

Shape averaging using Longest Common Sub-Sequence(LCSS): The longest common subsequence (Bergroth & Hakonen, 2000) generally permits to make a summary of a set of sequences. This approach supports the elastic distances and unequal size time-series. Aghabozorgi et al. (2011) and Aghabozorgi, Wah, Amini, and Saybani (2012) propose a fuzzy clustering approach for time-series clustering, and utilize the averaging method by LCSS as prototype.

2.6.3 Using Local Search Prototype

In this approach, at first the medoid of cluster is computed. Then, using averaging method (Section 2.6.2), averaged prototype is calculated based on warping paths. Next, new warping paths are calculated to the averaged prototype. Hautamaki et al. (2008) propose a prototype obtained by local search, instead of medoid to overcome the poor quality in time-series clustering in Euclidean space. They apply medoid, average and local search on k-Medoids, Random Swap (RS) and Agglomerative Hierarchical clustering (where k-means is used to fine-tune the output) to evaluate their work. They figure out that local search provides the best clustering accuracy and also more improvement to k-Medoids. However, it is not clear how much improvement it has in comparison with other works such as medoid averaging methods which are another frequently used prototype.

2.6.4 Discussion

One of the problems which lead to low accuracy of clusters is poor definition or updating method of prototypes in time-series clustering process, especially in partitioning approaches. Many clustering algorithms suffer from low accuracy of representation methods (Hautamaki et al., 2008; Niennattrakul & Ratanamahatana, 2007b). Moreover, the inaccurate prototype can affect convergence of clustering algorithms which results in low quality of obtained clusters (Niennattrakul & Ratanamahatana, 2007b). Different approaches of defining prototypes were discussed in Section 2.6. In this study, the averaging approach is used in order to find the prototypes of the sub-clusters (see Section 4.4.3) because the used distance metric is a none-elastic distance metric (ED). For the merging purpose, however, an arbitrary method can be used if it is compatible with elastic methods such as (Petitjean et al., 2011) for different schemes (see Section 4.5.2), however the simple "medoid" is used as prototype to be compatible with the elasticity of distance metric DTW, with k-Medoids algorithm, and also to provide fair condition for evaluation of the propose model with existing approaches.

2.7 Evaluation Measure

As regards the time-series clustering algorithms, the evaluation measures employed in the different approaches are discussed in this section. Visualization and scalar measurements are the major technique for evaluation of clustering quality which also is known as clustering validity in some articles (Hathaway & Bezdek, 2003). The techniques used to evaluate the proposed model in this study are explained in the following sections as depicted in Figure 2.11.

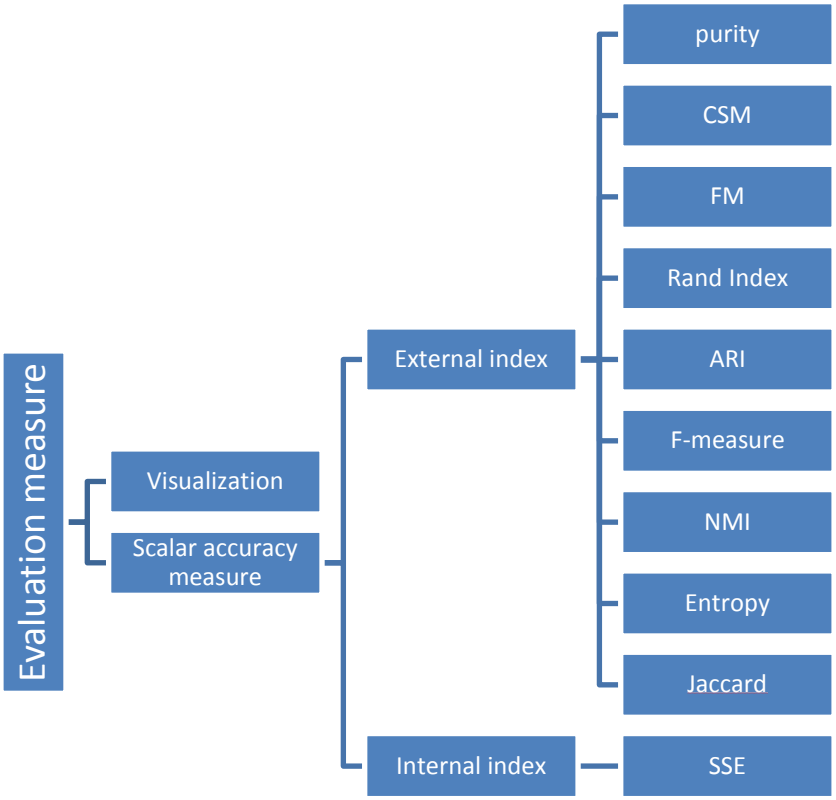


Figure 2.11: Evaluation measure hierarchy used in the literature

In scalar accuracy measurements, a single real number is generated to represent the accuracy of different clustering methods. Numerical measures that are applied to judge various aspects of cluster validity are classified into the following two types.

External Index: This index is used to measure the similarity of formed clusters to the externally supplied class labels or ground truth, and is the most popular clustering evaluation method (Halkidi, Batistakis, & Vazirgiannis, 2001). In the literature, this index is known also as external criterion, external validation, extrinsic methods, and supervised methods because the ground truth is available.

Internal Index: This index is used to measure the goodness of a clustering structure without respect to external information. In the literature, this index is known also as internal criterion, internal validation, intrinsic and unsupervised methods.

2.7.1 External Index

External validity indices are the measures of the agreement between two partitions, one of which is usually a known/golden partition (ground truth), e.g., true class labels, and another is from the clustering procedure. Ground truth is the ideal clustering that is often built using human experts. In this type of evaluation, ground truth is available, and the index evaluates how well the clustering matches the ground truth (Manning, Raghavan, & Schutze, 2008). Complete reviews and comparisons of some popular techniques exist in the literature (Amigó, Gonzalo, Artiles, & Verdejo, 2009; Gan & Ma, 2007; Meila, 2003; Rosenberg & Hirschberg, 2007). However, there is not a compromise and universally accepted technique to evaluate clustering approaches, though there are many candidates which can be discounted for a variety of reasons.

For external indices, usually match corresponding clusters and information theoretic are used as approach. Based on these approaches, many indices are presented in different articles (Amigó et al., 2009; Kremer et al., 2011); however, for readability and space reasons, the indices which have been used for evaluation of time-series clustering in the literature. Rand index, Adjusted Rand index, Purity, Jaccard index, Fowlkes-Mallows (FM), F-measure, Entropy, Normalized Mutual Information, and Cluster Similarity Measure are used in the experiments in this study (see Chapter 6.0).

Cluster purity: One of the ways of measuring the quality of a clustering solution is cluster purity (Zhao & Karypis, 2004). *Purity* is a simple and transparent evaluation measure. Considering $G = \{G_1, G_2, \dots, G_M\}$ as ground truth clusters, and $C = \{C_1, C_2, \dots, C_M\}$ as the clusters made by a clustering algorithm under evaluations, in order to compute the purity of cluster C with respect to G, each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned objects and dividing by number of objects in the cluster. Let there be k clusters (e.g., the k in k-Means) in the dataset D

and size of cluster C_j is shown by $|C_j|$. Let $\max(|C_j \cap G_i|)$ denote number of items of class G_i assigned to cluster C_j. Purity of the cluster C_j is given by:

$$purity(C_j) = \frac{1}{|C_j|}\max(|C_j \cap G_i|) \quad 2.5$$

Then, the overall purity of a clustering solution could be expressed as a weighted sum of individual cluster purities:

$$purity(C, G) = \sum_{j=1}^{k} \frac{|C_j|}{|D|} purity(C_j) \quad 2.6$$

Bad clusterings have purity values close to 0, and a perfect clustering has a purity of 1. However, high purity is easy to achieve when the number of clusters is large, in particular, purity is 1 if each objects gets its own cluster. Thus, one cannot only rely on purity as the quality measure. Purity was used for evaluation of time-series clustering in different studies (Ratanamahatana & Niennattrakul, 2006; X. Wang et al., 2006).

Cluster Similarity Measure (CSM): CSM (Warrenliao, 2005) is a simple metric used for validity of clusters in time-series domain (Kalpakis et al., 2001; J. Lin, Vlachos, et al., 2004; Xiong & Yeung, 2004; H. Zhang et al., 2006). Cluster similarity measure of a cluster C_j is given by:

$$CSM(C, G) = \frac{1}{k}\sum_{i=1}^{k} Max_{1 \le j \le k}\, Sim(C_j, G_i) \quad 2.7$$

where

$$Sim(C_j, G_i) = \frac{2|C_j \cap G_i|}{|C_j| + |G_i|} \quad 2.8$$

Folkes and Mallow index (FM): To calculate the FM, at first, the following quantities are considered as true clustering (ground truth) and errors in clustering process:

Let |TP| (True Positive) be the number of pairs, each belongs to one class in G (ground truth) and are clustered together in C. The |TN| (True Negative) is the number of pairs, each neither belongs to the same class in G, nor clustered together in C. Then, the error clusterings are the |FN| (False Negative) which is the number of pairs that are belong to one class in G, but are not clustered together in C, and |FP| (False Positive) which is the number of pairs that are not belong to one class in G (dissimilar objects), but are clustered together in C. Then the FM measure (Fowlkes & Mallows, 1983) is defined as:

$$FM(C,T) = \sqrt{\frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FN}|} \cdot \frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FP}|}} \qquad 2.9$$

This metric is the index for computing the accuracy of time-series clustering in multimedia domain (Ratanamahatana et al., 2005; H. Zhang et al., 2006).

Jaccard Score: Jaccard (Fowlkes & Mallows, 1983) is one of the metrics that has been used in various studies as external index (Chiş et al., 2009; Ratanamahatana et al., 2005; H. Zhang et al., 2006). Considering the parameters defined for FM index, the jaccard index is defined as:

$$Jaccard(C,T) = \sqrt{\frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FP}| + |\mathrm{FN}|}} \qquad 2.10$$

Rand index (RI): A popular quality measure (Chiş et al., 2009; Ratanamahatana et al., 2005; H. Zhang et al., 2006) for evaluation of time-series clusters is the Rand index (Rand, 1971; J. Wu, Xiong, & Chen, 2009), which measures the agreement between two partitions, that is, how the clustering results are close to the ground truth. The agreement between C and G can be estimated using:

$$RI(C,G) = \sqrt{\frac{|\mathrm{TP}| + |\mathrm{TN}|}{|\mathrm{TP}| + |\mathrm{TN}| + |\mathrm{FP}| + |\mathrm{FN}|}} \qquad 2.11$$

Adjusted Rand Index (ARI): RI does not take a constant value (such as zero) two random clustering. Hence, in (Hubert & Arabie, 1985), authors suggest a corrected-for-chance version of the RI which works better than RI and many other indices (G. W. Milligan & Cooper, 1986; Steinley, 2004). This approach was used in gene expression domain successfully (Yeung, Fraley, Murua, Raftery, & Ruzzo, 2001; Yeung, Haynor, & Ruzzo, 2001). In this approach the expected RI of random labelling, i.e.E[RI], is discounted as:

$$ARI(C,G) = \frac{\mathrm{RI} - \mathrm{E[RI]}}{\max(RI) - \mathrm{E[RI]}} \quad 2.12$$

F-measure: F-measure (Van Rijsbergen, 1979) is a well-established measure for assessing the quality of any given clustering solution with respect to ground truth. F-measure ($F \in [0, 1]$) is defined based on precision and recall:

The precision of an object indicates how many items in the same cluster belong to the same class (ground truth) (Van Rijsbergen, 1979) which is estimated as:

$$Precison(C,G) = \sqrt{\frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FP}|}} \quad 2.13$$

The recall of an object reflects how many objects of the same class (in ground truth) are assigned to the same cluster (Van Rijsbergen, 1979):

$$Recall(C,G) = \sqrt{\frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FN}|}} \quad 2.14$$

Then, F-measure is calculated as the harmonic mean between precision (P) and recall (R):

$$F - measure(C,G) = \sqrt{\frac{2 \times Precison(C,T) \times Recall(C,T)}{Precison(C,T) + Recall(C,T)}} \quad 2.15$$

F-measure compares how closely each cluster matches a set of categories of ground truth. F-measure has been used in clustering of time-series data (Chiş et al., 2009; Gullo et al., 2011; Kameda & Yamamura, 2006; Van Rijsbergen, 1986) and in natural language processing for evaluating clustering (Larsen & Aone, 1999).

Normalized Mutual Information (NMI): As mentioned, high purity in the large number of clusters is a drawback of purity measure. In order to make trade-off between the quality of the clustering against the number of clusters, NMI (Studholme, Hill, & Hawkes, 1999) is utilized as quality measure. Next, it has been utilized in various studies (Fern & Brodley, 2004; Strehl & Ghosh, 2003; H. Zhang et al., 2006). Moreover, NMI can be used to compare clusterings with different numbers of clusters, because this measure is normalized (Manning et al., 2008).

$$NMI(C,G) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k}|C_j \cap G_i| * \log(\frac{|D| * |C_j \cap G_i|}{|C_j| * |G_i|})}{\sqrt{\left(\sum_{i=1}^{k}|G_i| * \log\frac{|G_i|}{|D|}\right) * (\sum_{j=1}^{k}|C_j| * \log\frac{|C_j|}{|D|})}} \quad 2.16$$

Entropy: Entropy (S. Lin, Song, & Zhang, 2008; Rohlf, 1974) of a cluster shows how dispersed classes are with a cluster (this should be low). Entropy is a function of the distribution of classes in the resulting clusters. In the case of entropy, for each cluster C_j, the class distribution of data is computed as the probability $\Pr(G_i|C_j)$ that an instance in C_j belongs to class G_i. Using this class distribution, the normalized entropy of C_j is computed as:

$$Entropy(C_j) = -\frac{1}{\log h}\sum_{i=1}^{h}\Pr(G_i|\,C_j) \times \log(C_i|C_j)) \quad 2.17$$

where $\Pr(G_i|C_j) = |C_j \cap G_i|/|C_j|$. The overall entropy ($E \in [0, 1]$) is defined as the sum of the individual cluster entropies weighted by the size of each cluster:

$$Entropy(C, G) = \frac{1}{|D|}\sum_{j=1}^{K} |C_j| \times Entropy(C_j) \quad 2.18$$

Based on the above measures, a good clustering solution is expected to have both high F-measure and low entropy. This metric was used for evaluation of clustering time-series in the literature (Gullo et al., 2011; Van Rijsbergen, 1986).

In short, one of the most popular approaches for quality evaluation of clusters is external indices to find how good the finding cluster results are (Halkidi et al., 2001) which also is used for evaluation of the proposed models in this study. However, it is not directly applicable in real-life unsupervised tasks, because the ground truth is not available for all datasets. Therefore, in the case that ground truth is not available, internal index is used (see Section 6.2.7 where internal index is used for evaluation).

2.7.2 Internal Index

Typical objective functions in clustering, formalize the goal of attaining high intra-cluster similarity (objects within a cluster are similar) and low inter-cluster similarity (objects from different clusters are dissimilar). Internal validation compares solutions based on the goodness of fit between each clustering and the data. Internal validity indices evaluate clustering results by using only features and information inherent in a dataset. They are usually used in the case that true solutions (ground truth) are unknown. However, this index can only make comparisons between different clusterings generated using the same model/metric. Otherwise, it makes assumptions about cluster structure.

There are many internal indices such as Sum of Squared Error, Silhouette index, Davies-Bouldin, Calinski-Harabasz, Dunn index, R-squared index, Hubert-Levin (C-index), Krzanowski-Lai index, Hartigan index, Root-Mean-Square Standard Deviation

(RMSSTD) index, Semi-Partial R-squared (SPR) index, Distance between two clusters (CD) index, Weighted inter-intra index, Homogeneity index, and Separation index.

Sum of Squared Error (SSE): SSE is an objective function that describes the coherence of a given cluster, "better" clusters are expected to give lower SSE values (Han & Kamber, 2011). For evaluation of clusters in terms of accuracy, the Sum of Squared Error (SSE) can be used as the most common measure in different works (J. Lin, Vlachos, et al., 2004; Vlachos et al., 2003). For each time-series, the error is the distance to the nearest cluster. To get SSE, the following formula is used:

$$\mathrm{SSE} = \sum_{j=1}^{c} \sum_{F_i \in C_j} (\mathrm{dis}(F_i, R_j))^2 \qquad 2.19$$

where, F_i is a data point in cluster C_j , and R_j is the representative (prototype) for cluster C_j.

2.8 Related works: Time-series Clustering Algorithms

There are many articles related to different approaches of clustering in the literature (Berkhin, 2006; Jain et al., 1999; Rauber, Pampalk, & Paralič, 2000; Xu & Wunsch, 2005). However, the number of the researches about the time-series clustering is quite scarce compared with those works which have focused on static data, though, literature trend shows a trend of increased activity.

In this section, the existing works related to clustering of time-series data are concentrated and discussed. Some of them are using raw time-series and some try to use reduction methods before clustering of time-series data. In general, clustering in its conventional form can be broadly classified into five groups (Han & Kamber, 2011): Partitioning, Hierarchical, Grid-based, Model-based and Density-based clustering algorithms. In the following, the application of each group in time-series clustering is discussed in detail.

2.8.1 Hierarchical Clustering of Time-series

Hierarchal clustering (Kaufman et al., 1990) is an approach of cluster analysis which makes a hierarchy of clusters using agglomerative or divisive algorithms. Agglomerative algorithm considers each item as a cluster, and then gradually merges the clusters (bottom-up). In contrast, divisive algorithm starts with all objects as a single cluster and then splits the cluster to reach the clusters with one object (top-down). In general, hierarchical algorithms are weak in terms of quality because they cannot adjust the clusters after splitting a cluster in divisive method, or after merging in agglomerative method. As a result, usually hierarchical clustering algorithms are combined with another algorithm as a hybrid clustering approach to remedy this issue. Moreover, some extended works are done to perform the performance of hierarchical clustering such as Chameleon (Karypis, Han, & Kumar, 1999), CURE (Guha, Rastogi, & Shim, 1998) and BIRCH (T. Zhang, Ramakrishnan, & Livny, 1996) where the merge approach is enhanced or constructed clusters are refined.

In hierarchical clustering of time-series also, nested hierarchy of similar groups is generated based on a pair-wise distance matrix of time-series (Vlachos et al., 2003). Hierarchical clustering has a great visualization power in time-series clustering (Keogh & Pazzani, 1998; Van Wijk & Van Selow, 1999). This characteristic of hierarchical clustering leads to be used for time-series clustering to a great extent. For example, Oates, Schmill, and Cohen (2000) use agglomerative clustering to produce the clusters of the experiences of an autonomous agent. They use Dynamic Time Warping (DTW) as a dissimilarity measure with a dataset containing 150 trials of real Pioneer data in a variety of experiences. In another study by Hirano and Tsumoto (2005), the authors use average linkage agglomerative clustering which is a type of hierarchical approach for time-series clustering. Moreover, in many researches, hierarchical is used to evaluate dimensionality reduction or distance metric due to its power in visualization. For

example, in a study (J. Lin, Keogh, Lonardi, et al., 2003), the authors present Symbolic Aggregate Approximation (SAX) representation and use hierarchical clustering to evaluate their work. They show that using SAX, hierarchical clustering has a result similar with Euclidean distance.

Additionally, in contrast to most algorithms, hierarchy clustering does not require the number of clusters as an initial parameter which is a well-known and outstanding feature of this algorithm. It is also a strength point in time-series clustering, because usually it is hard to define the number of clusters in real world problems.

Moreover, despite many algorithms, hierarchical clustering has the ability to cluster time-series with unequal length. It is possible to cluster unequal time-series using this algorithm if an appropriate elastic distance measure such as Dynamic Time Warping (DTW) (Sakoe & Chiba, 1971, 1978) or Longest Common Subsequence (LCSS) (Banerjee & Ghosh, 2001; Vlachos et al., 2002) is used to compute the dissimilarity/similarity of time-series. In fact, lack of necessity for prototypes in its process, has made this algorithm capable to accept unequal time-series.

However, hierarchical clustering is essentially not capable to deal effectively with large time-series (X. Wang et al., 2006) due to its quadratic computational complexity. Accordingly, it leads to be restricted to small datasets because of its poor scalability.

2.8.2 Partitioning Clustering

A partitioning clustering method, makes *k* groups from *n* unlabelled objects such that each group contains at least one object. One of the most used algorithms of partitioning clustering is **k-Means** (MacQueen, 1967) where each cluster has a prototype which is the mean value of its objects. The main idea behind k-Means clustering is the minimization of the total distance (typically Euclidian distance) between all objects in a cluster from their cluster center (prototype). Prototype in k-Means process is defined as

mean vector of objects in a cluster. However, when it comes to time-series clustering, it is a challenging issue and is not trivial (Niennattrakul & Ratanamahatana, 2007b). Another member of partitioning family is **k-Medoids (PAM)** algorithm (Kaufman et al., 1990), where the prototype of each cluster is one of the nearest objects to the centre of the cluster. Moreover, CLARA and CLARANS (Ng & Han, 1994) are improved version of k-Medoid algorithm for mining in spatial databases. In both k-Means and k-Medoids clustering algorithms, number of clusters, *k*, has to be pre-assigned, which is not available or feasible to determine for many applications, so it is impractical in obtaining natural clustering results and is known as one of their drawbacks in static objects (X. Wang et al., 2006) and also time-series data (Antunes & Oliveira, 2001). It is more crucial in time-series because the datasets are very large and diagnostic checks for determining the number of clusters is not easy. Accordingly, authors in (Fayyad et al., 1998) investigate the role of choosing correct initial clusters in quality and time-execution of k-Means in time-series clustering.

However, k-Means and k-Medoids are very fast compared to hierarchical clustering (Bradley, Fayyad, & Reina, 1998; MacQueen, 1967) and it has made them very suitable for time-series clustering and has been used in many works (Bagnall & Janacek, 2005; Beringer & Hullermeier, 2006; C. Guo et al., 2008; Hautamaki et al., 2008; J. Lin, Vlachos, et al., 2004). As a specific case, authors in (Ratanamahatana & Niennattrakul, 2006), use the robustness of k-Medoids to noise and outliers, in order to cluster time-series of multimedia data.

k-Means and k-Medoids algorithms make clusters which are constructed in 'hard' or "crispy" manner, that is, an object either is or is not a member of a cluster. On the other hand, **FCM** (**Fuzzy c-Means)** algorithm (Bezdek, 1981; Dunn, 1973) and Fuzzy c-Medoids algorithm (Krishnapuram, Joshi, Nasraoui, & Yi, 2001) build 'soft' clusters. In fuzzy clustering, an object has a degree of membership in each cluster (Dembélé &

Kastner, 2003). Fuzzy partitioning algorithms have been used for time-series clustering in some areas. For example, in (Tran & Wagner, 2002), authors use FCM (Fuzzy c-Means) to cluster time-series for speaker verification. In another work (Alon & Sclaroff, 2003), the authors use fuzzy variant to cluster similar object motions that were observed in a video collection. They adopt an EM-based algorithm and a mixture of HMMs to cluster time-series data. Then, each time-series is assigned to each cluster to a certain degree. Moreover, using FCM, authors in (Golay et al., 1998) cluster MRI time-series of brain activities. They use raw univariate time-series of equal length. As distance metric, they use Euclidian distance and cross-correlation. They evaluate their work with different numbers of clusters (k) and recommend using a large number of clusters as initial clusters. However, it is not defined how they achieve the optimal number of clusters in this work.

Generally, partitioning approaches, whether crispy or hard, need defining prototypes and their accuracy are directly depends on the definition of prototypes and updating method. Hence, they are more compatible with finding clusters of similar time-series in time (preferably equal length time-series) because defining the prototype for elastic distance measures (which handle the similarity in shape) is not very straight forward, as was discussed in Section 2.6.

2.8.3 Model-based Clustering

Model-based clustering tries to recover the original model from a set of data. This approach assumes a model for each cluster, and finds the best fit of data to that model. In detail, it presumes that there are some centroids chosen at random, and then some noise is added to them with a normal distribution. The model that is recovered from the generated data defines clusters (Shavlik & Dietterich, 1990). Typically, model-based methods use either statistical approaches, e.g., COBWEB (Fisher, 1987), or Neural

Network approaches, e.g., ART (Carpenter & Grossberg, 1987) or Self-Organization Map (Kohonen, 1990).

In some of works in time-series clustering area, authors use Self-Organizing Maps (SOM) for clustering of time-series data. As mentioned, SOM is a model-based clustering based on neural networks, which look likes processing that happens in the brain. For example, in (X. Wang et al., 2004), authors use SOM to cluster time-series features. However, because SOM needs to define the dimension of weight vector, it cannot work well with time-series of unequal length (Warrenliao, 2005).

Additionally, there are a few articles which use model based clustering of time-series data which are composed of polynomial models (Bagnall & Janacek, 2005), Gaussian mixed models (Biernacki, Celeux, & Govaert, 2000), ARIMA (Corduas & Piccolo, 2008) , Markov chain (Ramoni et al., 2000) and Hidden Markov models (Bicego, Murino, & Figueiredo, 2003; Hu et al., 2006). In general, model based clustering has two drawbacks: first, it needs to set parameters and it is based on user assumptions which may be false and result in inaccurate clusters. Second, it has a slow processing time (especially neural networks) on large datasets (Andreopoulos, An, & Wang, 2009).

2.8.4 Density-based Clustering

In density based clustering, clusters are subspaces of dense objects which are separated by subspaces in which objects have low density. One of the famous algorithms which works by density-based concept is DBSCAN (Ester, Kriegel, & Sander, 1996) where a cluster is expanded if its neighbors are dense. OPTICS (Ankerst, Breunig, & Kriegel, 1999) is another density-based algorithm which addresses the issue of detecting meaningful clusters in data of varying density. The model proposed by Chandrakala and Chandra (2008) is one of the rare cases, where the authors propose a density based clustering method in kernel feature space for clustering multivariate time-series data of

varying length. Additionally they present a heuristic method of finding the initial values of the parameters used in their proposed algorithm. However, density-based clustering has not been used broadly for time-series data clustering (in the literature), because of its rather high complexity.

2.8.5 Grid-based Clustering

The grid-based methods quantize the space into a finite number of the cells that form a grid, and then perform clustering on the grid's cells. STING (W. Wang, Yang, & Muntz, 1997) and Wave Cluster (Sheikholeslami, Chatterjee, & Zhang, 1998) are two typical examples of clustering algorithms which are based on grid-based concept. To the best of our knowledge, there is no work in the literature applying grid-based approaches for clustering of time-series.

In Table 2.4 a summary of related works are mentioned based on the adopted representation method, distance measure, clustering algorithm and definition of prototype (if it is applicable).

Table 2.4: Whole time-series clustering algorithms

Article	Representation method	Distance measurement	Prototype	Clustering algorithm	TS Size	Noise robustness	Unequal time-series	Comments (P:Positive, N:Negative)	Application
Hautamaki et al.(2008)	Raw time-series	DTW	New prototype	K-mean, Hierarchical, RS	-	-	Yes	P: Only was compared with medoid	-
Gullo, Ponti, Tagarelli, Tradigo, & Veltri (2011)	DSA	DTW	A summarization approach in (Gullo, Ponti, Tagarelli, & Greco, 2009)	K-Means	-	-	NO	-	Mass spectrometry clustering
Möller-Levet, Klawonn, Cho, & Wolkenhauer (2003)	piecewise linear function	STS	Yes	Modified FCM	Short	-	NO	-	Biology, DNA microarray
(X. Zhang et al., 2011)	Raw time-series	triangle distance	N/A	Hierarchical	-	-	NO		
Bao (2007) Bao & Yang (2008)	a critical point model (CPM)	-	N/A	turning points	-	-	NO	P: Using important points	Financial
Fu, Chung, Luk, & Ng (2010)	PIP (perceptually important points)	Vertical distance	estimated means	k-Means	-	No	Yes	P: incremental N: Only indexing	Financial, time-series query
Lin, Vlachos, Keogh, & Gunopulos (2004)	Wavelets.	Euclidean Distance	Averaging	partitioning clustering, k-Means and EM	-	No	No	P: Incremental	-
Vlachos, Lin, & Keogh (2003)	DWT (Discrete Wavelet Transform) Haar wavelet	Euclidean	Not clear	k-means,	-	No	No	P: Incremental	-
X. Wang, Smith, & Hyndman (2005)	global characteristics	Euclidean	-	SOM	Long	No	NO	N: Only focus on dimensionality reduction method	-
Ratanamahatana, Keogh, Bagnall, & Lonardi (2005)	BLA (clipped time-series representation)	LB_clipped	Not clear	k-means	Long	No	No	-	-
Focardi & others (2005)	Raw time-series	3 types of distances	-	-	-	No	NO	N: Using Raw time-series	-
Lin, Keogh, Wei, & Lonardi (2007)	ESAX	Min-Distance	Not clear	Partitioning Hierarchal	-	No	No	N: Only focus on distance measurement	-
Abonyi, Feil, Nemeth, & Arva (2005)	PCA	SpCA Factor	N/A	Hierarchical	-	No	No	P: Anomaly detection	Multivariate data
Z. J. Wang & Willett (2004)	Raw time-series	GLR (generalized likelihood ratio)	N/A	two stages approach	-	No	NO	N: Subsequence Segmentation	-
Qian, Dolled-Filhart, Lin, Yu, & Gerstein (2001)	Raw time-series	Ad hoc distance	N/A	Single-linkage	-	No	No	N: using raw time-series	Gene expression
Tseng & Kao (2005)	gene expression	Euclidean distance, Pearson's correlation	N/A	Modified CAST	-	No	No	P: Focus on clustering	-

Golay et al.(1998)	Raw time-series	Euclidean & two cross correlation-based	N/A	FCM	-	Yes	NO	-	brain activity
Bagnall & Janacek (2005)	Clipped	Euclidean	Medoid	k-Means, k-Medoids	Short, long	N/A	No	-	-
Kakizawa, Shumway, & Taniguchi (1998)	Raw time-series	J divergence	N/A	Agglomerative hierarchical	-		No	P: Multiple variable support	Earthquake
Košmelj & Batagelj (1990)	Raw time-series	Euclidean	N/A	Modified relocation clustering	-	-	No	P: Multiple variable support	Commercial energy consumption
Kumar & Patel (2002)	Raw time-series	Gaussian models of data errors	N/A	Agglomerative hierarchical	-	-	No	-	Seasonality pattern in retails
Liao (2005)	SAX	Euclidean and symmetric version of Kullback–Liebler	N/A	*k*-Means and fuzzy *c*-Means	-	-	Yes	P: Multiple variable support	Battle simulations
Liao et al.(2002)	Raw time-series	DTW and Kullback–Liebler distance	N/A	*k*-Medoids-based genetic clustering	-	No	Yes	N: Single variable support	Battle simulations
Policker & Geva (2000)	Raw time-series	Euclidean	N/A	Fuzzy clustering	-	-	No	N: Single, using raw time-series	Sleep EEG signals
Shumway (2003)	Raw time-series	Kullback–Leibler discrimination information Measures	N/A	Agglomerative hierarchical	-	-	No	P: Multiple variable support	Earthquakes and mining explosions
Van Wijk & Van Selow (1999)	Raw time-series	Root mean square	N/A	Agglomerative hierarchical	-	-	No	N: Single variable, using raw time-series	Daily power consumption
Wismüller et al.(2002)	Raw time-series	N/A	N/A	Neural network clustering	-	-	No	N: Single variable support, using raw time-series	Functional MRI brain activity mapping
Liu & Shao (2009)	SAX	trend statistics distance	N/A	Hierarchical	-	-	NO	P: Using symbolized TS	New similarity distance
Guo, Jia, & Zhang (2008)	feature-based using ICA	-	Not clear	modified k-means	-	No	NO	-	stock time-series
Lai, Chung, & Tseng (2010)	SAX, Raw time-series	Min-Dist, Eucleadian distance	-	two-level clustering: CAST,CAST	-	No (DWT)	Yes	N: Based on subsequence ,CAST is poor in front of huge data	Gene expression
Ratanamahatana & Niennattrakul (2006)	Raw time-series	Dynamic Time Warping	Medoid	k-Means, k-Medoids	-	Yes	No	N: using raw time-series	Multimedia time-series
(Keogh, Lonardi, et al., 2004)	SAX	compression-based distance	-	Hierarchy	-	No	No	-	-

Considering many works, it was understood that in most of models, theauthors use time-series data as raw data or dimensionality reduced data, with standard traditional clustering algorithms. It is obvious that this type of analyzing time-series which use a brute-force approach without any optimization is a proper solution for scientific theories, but not for real world problems, because they are naturally very slow or inaccurate in large data bases. As a result, in many studies the attention of the researchers has drawn to using more customized algorithms for time-series data clustering as the ultimate solution.

In the following section, focusing on the algorithm, specific approaches are discussed where the emphasize is on the solutions which address the low quality of time-series clustering problem due to mentioned issues in process of clustering. This section can be considered crucial to this thesis.

2.8.6 Multi-step Clustering

Although there are many studies to improve the quality of representation approaches, distance measurement, and prototypes, a few articles emphasis on enhancing algorithms and present a new model (usually as a hybrid method) for clustering of time-series data. In the following the most related works are presented and discussed:

Lai et al. (2010) describe the problem of overlooking of information using dimension reduction. They claim that overlooked information could provide different meaning in time-series clustering results. To solve this issue, they adopt a two-level clustering method, where both the whole time-series and the subsequence of time-series are taken into account in the first and second level respectively. They used SAX transformation as dimension reduction method and CAST as clustering algorithm in the first level in order to group first-level data. In the second level, to measure distances between time-series, Dynamic Time Warping (DTW) has been used for varying length data, and Euclidean

distance for equal length data. Finally, second-level data, of all the time-series, are then grouped by a clustering algorithm.

Discussion:

1) The distance measure method used in order to find the first level result, is not clear while it is of great importance, because, for example, if the length of time-series are different (which is a possible case), it will effect on choosing dimension reduction and distance measurement methods.
2) The authors have used CAST algorithm in their proposed approach for two times, once for making initial clusters, then for splitting each cluster into sub-clusters (although they used it 3 times in pseudo code). However, using CAST algorithm needs determining the threshold of affiliation which is a very sensitive parameter in this algorithm (Bellaachia, Portnoy, Chen, & Elkahloun, 2002).
3) In this work, more granulated time-series are clustered which is actually based on the sub-sequence clustering. However, the work done by Keogh and Lin (2005) indicates that subsequence clustering is meaningless. The authors in that work define "meaningless" as when the clustering output is independent of the input.
4) Their experimental result is not based on the published datasets in the literature. Therefore, there is not a way to compare their method with existing approaches for time-series clustering.

The authors in (X. Zhang et al., 2011) also propose a new multi-level approach for shape based time-series clustering. In the first step, some candidate time-series are chosen from a made one-nearest neighbour network. In order to make the network of time-series, authors propose triangle distance for calculating similarity between time-series data. Then, hierarchical clustering is performed on chosen candidate time-series. To handle the shifts in time-series, Dynamic Time Warping (DTW) is utilized in the

second step of clustering. Using this approach the size of data is reduced by approximately ten per cent.

Discussion:

1) This algorithm needs a nearest-neighbor network in the first level while complexity of making the nearest-neighbor network is O(n^2) which is very high. As a result, they try to reduce the search area by pre-clustering of data (using k-Means) and limit the search only in each cluster to reduce the cost of creation network. However, because raw time-series is used in the process of pre-clustering to reduce the size of data, making the network itself is still very costly. As a result, the complexity of whole clustering is high which is not applicable on large datasets.
2) Pre-clusters developed in this model may not be accurate because the pre-clusters are constructed by a non-elastic distance measure on raw time-series and it may be affected by outliers. Additionally, it is not clear how they solve the challenge of making the prototypes in k-Means while triangle is used as distance measure.
3) The experimental results are based on two syntactic datasets, however, the results should be tested on more datasets (Keogh & Kasetty, 2003) because characteristics of time-series varies in different data-sets from different domains.
4) The error rate of choosing the candidates is computed but the quality of the final clusters has not measured using any standard and common metrics to be comparable with other methods.

In a group of works, an incremental clustering approach is adopted which exploit the multi-resolution characteristic of time-series data to cluster them in multi-step. Vlachos et al. (2003) developed a method based on standard k-Means and Discrete Wavelet Transform (DWT) decomposition to cluster time-series data. They extended the k-

Means algorithm to perform clustering of time-series incrementally at different resolutions of DWT decomposition. At first, they use Haar wavelet transformations to decompose all the time-series. After that, they apply the k-Means clustering on various regulations from a chaos to a finer level. At the end of each level, the extracted centers are reused as the initial centers for the next level of resolution. They doubled the center coordinates of each level because the length of a time-series is doubled in next level. In this algorithm, more and more detail are used during the clustering process. In order to compute the clustering error, they computed clustering error at the end of each level by summing up the number of objects clustered incorrectly divided by the cardinality of the dataset. In another similar work, Lin et al. (2004) generalized this work and presented an anytime version of the partitioned clustering algorithm (k-mean and EM) for time-series. In this method also, authors use the multi-resolution property of wavelets in their algorithm. Following these works, Lin et al. in (J. Lin et al., 2005) present a multi-resolution clustering approach based on multi-resolution PAA (MPAA) for the incremental clustering algorithm of time-series.

Discussion:

1) In terms of speed of clustering these approaches are quite good, however, in all these models, it is not clear that to what level it should be continued (the termination point).
2) Additionally, in each iteration, all the time-series (which are in the same resolution) are re-clustered again. Therefore, the noise in some of them can affect the whole process.
3) Moreover, this model is applicable only for partitioning clustering, which implies that it is not working for other types of algorithms such as arbitrary shape algorithms or hierarchical algorithms in the case where user needs the structure of data (the hierarchy of clusters).

4) Another problem which these models should resolve is working with distance measures such as DTW which at first, are very costly and cannot be applied on whole dataset, and secondly, defining the prototypes using them is not a trivial task.

2.9 Chapter Summary

Although different researches have been carried on time-series clustering, the unique characteristics of time-series lead to most conventional clustering algorithms to not work well for time-series. In particular, the high dimensionality, very high feature correlation, and the (typically) large amount of noise that characterize time-series data have been viewed as an interesting research challenge in time-series clustering. Accordingly, most of the studies in the literature have concentrated on two subroutines of clustering:

1) A vast number of researches have focused on high dimensional characteristic of time-series data and tried to present a way of representing time-series in a lower dimension compatible with conventional clustering algorithms.

2) Different efforts have been taken on presenting a distance measurement based on raw time-series or the represented data.

The common characteristic in both above approaches is clustering of the transferred, extracted or raw time-series using conventional clustering algorithms such as k-Means, k-Medoid or hierarchical clustering (as it was discussed in 2.3). However, most of them suffer from overlooking of data (caused by dimensionality reduction), inaccurate similarity calculation (due to high complexity of accurate measures), and lack of quality in clustering algorithms (because of their nature which is suitable for static data).

Actually, considering literatures, it can be concluded that most of the studies are focusing on improving representation methods and distance measurement methods, and

the portion of enhancing clustering approaches is very small (Table 2.2, Table 2.3 and Table 2.4,).

Among a few approaches and algorithms which have been proposed for time-series clustering, there are some studies who have taken explicit or implicit strategies for increasing the quality (considering the scalability) in time-series clustering. However, one still can see the problem of low quality or lack of meaningfulness in the clusters. That is, clusterings are either accurate which are constructed expensively, or inaccurate but made inexpensively. Our intention in this study is to develop a flexible and accurate clustering model dedicated for clustering of large time-series datasets. In the following chapter, the methodology of this study to develop such an effective clustering model is explained.

3.0 RESEARCH METHODOLOGY

3.1 Introduction

This chapter explains the research methodology used in the study. The sub-topics in this chapter include an overview of the proposed models, the motivation for designing a multi-step clustering approach, and description of the methods used in this study to achieve the research objectives, mentioned in Chapter 1. Additionally, the approach used for evaluation of the model and methods are presented. The last section concludes this chapter with a chapter summary.

3.2 Approaches to Research

The research methodology framework of this thesis is shown in Figure 3.1. Each stage of the methodology for this research is explained in the following sub-sections.

3.2.1 Reviewing Related Works

Based on reviewing the literature, the characteristics and features of various time-series clustering approaches were analysed. The analysis of existing approaches gives a wider perspective of the problems in time-series clustering. It had been stated that essentially four elements are essential in the clustering of time-series, i.e., distance measure, representation method, prototype definition, and clustering algorithm.

3.2.2 Problem Formulation

The literature review had clearly examined the issues in the clustering approaches. It has been found that in spite of the advances in representation methods and distance measures, the quality of clustering approaches is not high. Hence, the reasons of low quality in different approaches are investigated.

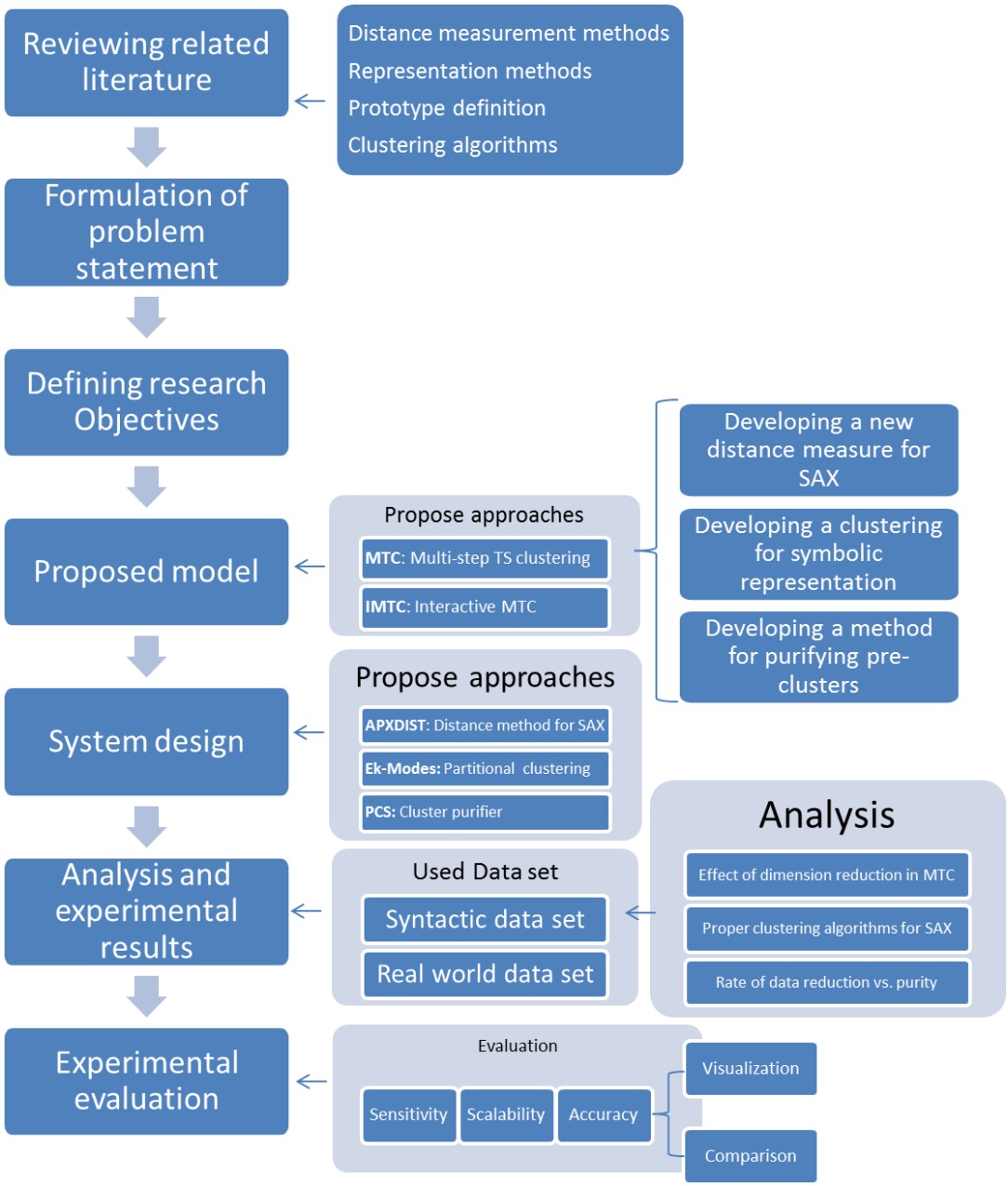


Figure 3.1: Research methodology framework

3.2.3 Definition of Research Objectives

Formulation of problem provides direction for this research to come up with the following objectives:

1. To propose and develop a clustering model to cluster large raw time-series data accurately. This objective, needs the following methods:
    a. To develop a distance measure for similarity calculation

b. To develop a clustering approach for approximate clustering of the time-series data transformed to symbolic representation.
c. To develop a method to dynamically split the pre-cluster to purer clusters

2. To extend the proposed model, enabling to run interactively
3. To evaluate the capability of the proposed models in improving the accuracy of clustering

To achieve these objectives two models are proposed and evaluated extensively in this study. The following section explains briefly about the proposed models.

3.2.4 Proposed Models

To achieve the first objective, a multi-step approach namely MTC, is proposed. The motivation for using a multi-step approach is addressing the issues in the existing approaches. Then the model is extended (IMTC) to achieve the second objective.

3.2.4.1 MTC

MTC includes three steps: pre-clustering, purifying and merging. Figure 3.2 shows the overall view of the process in MTC briefly.

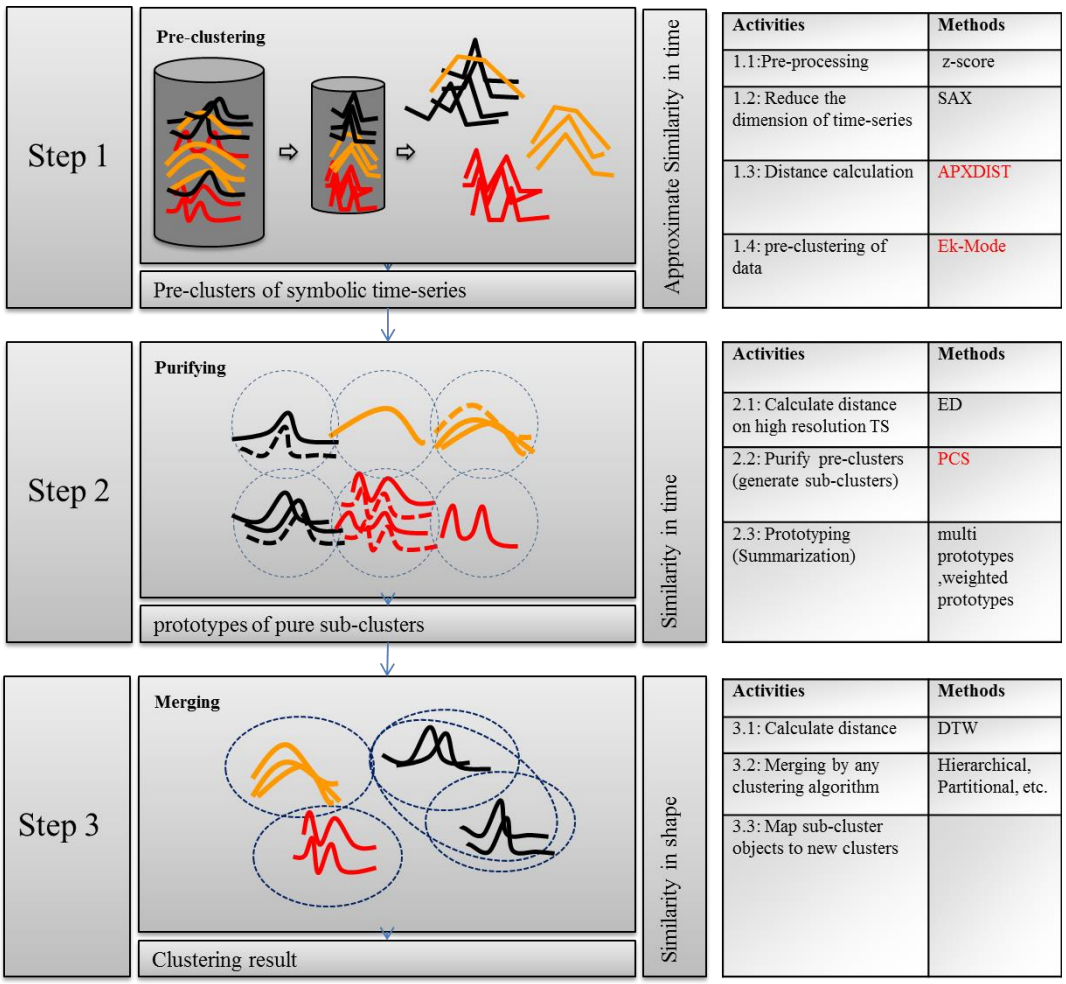


Figure 3.2: Proposed model for clustering of time-series data (MTC)

1. Pre-clustering step: At first, z-normalization is used to normalize all time-series data. Then, time-series data is used in low-resolution mode. That is, the SAX is adopted in order to reduce the dimension of raw time-series before clustering. Then, a proper similarity measure, approximated distance (APXDIST), and an extended k-Modes algorithm (Ek-Modes) are designed to provide approximate clusters. This step (pre-clustering step), reduces the input size for the next step, and can be performed incrementally (see Section 4.3 for more details).

2. Purification and summarization: In this step, time-series data are used in high-resolution mode (or higher resolution mode). In order to purify the pre-clusters, an affinity search technique (PCS) is designed for splitting of time-series data, which

creates sub-clusters. Then, the prototypes are generated for time-series which exist in the prepared sub-clusters (more details will be provided in Section 4.4).

3. Merging: In the third step, the algorithm goes bottom-up. Prototypes prepared in high-level step are utilized for merging. That is, a clustering algorithm is used to merge the prototypes which are much smaller than the original dataset. It leads to decrease in the number of iterations in this step, very fast convergence, thus low cost execution. Moreover, the final clusters can be sent into the second step (as pre-clusters) for increasing the quality incrementally. In this case, MTC performs as an interactive clustering approach (it is explained further in Section 4.7).

In the following, the motivation for the design of a multi-step approach is presented:

1. Motivation for step1 (pre-clustering step):
    - Data mining is constrained by disk I/O especially in large datasets because typically don't fit in main memory, and disk I/O tends to be the bottleneck for any data mining task (Faloutsos et al., 1994). Assume that you have one Gigabyte of main memory and want to do k-Means clustering. Then, clustering of 1 Gigabyte data may take a few minutes but clustering of 1.1 gigabytes of data, takes 20 hours (Keogh, 2007). The generic solution for this problem is to create an approximation of the data (Keogh & Pazzani, 2000; J. Lin, Keogh, Lonardi, et al., 2003), which will fit in main memory, yet retains the essential features of interest (discussed vastly in 2.4). As a result, the whole data can be loaded in main memory and the problem at hand is solved approximately (Bradley et al., 1998; Keogh, 2007).
    - Since time-series data are normally embedded by noise, dimensionality reduction results in noise shrinkage, which can improve the mining quality (H. Zhang et al., 2006). Accordingly, initializing the clusters on a low dimension

approximation of the data can improve the quality preventing the local minimum problem (C. Ding et al., 2002). It is the satisfactory motivation for using dimensionality reduced data as initial cluster in pre-clustering step (see Section 4.3.2).

- Partitioning clustering of objects is an appropriate choice when clusters are compact, rather equal size and well separated (Guha et al., 1998). Using a multi-step approach, the advantageous of partitioning clustering in splitting clusters is used in the pre-clustering (it is discussed further in 4.5).

2. Motivation for step 2 (purifying step):
    - Because of overlooking of data in the dimensionality reduction process, one cannot rely on the clustering results provided by dimension reduction approaches (especially in sensitive datasets). As a result, regardless of adopted techniques, the clustering should be applied on high-resolution data.
    - The number of time-series in the dataset (cardinality of dataset) is as important as length of time-series. An algorithm which can deal with a dataset with a large number of time-series is desirable. Therefore, reduction of data by defining representative (s) for a group of very similar time-series reduces the complexity of clustering algorithm.
3. Motivation for step 3 (merging step):
    - *Similarity in shape* is desirable in the clustering of time-series, however, the state-of-the-art methods for similarity evaluation of time-series are mostly quadratic because these methods use the Dynamic Programming method (Salvador & Chan, 2007). The cost of comparing two time-series using this technique is quadratic in the length of the time-series. This makes the measuring of similarity between two time-series very expensive. Although, many pruning techniques have been devised so that time-series similarity

queries do not need to compute the similarity measures between the query and every time-series in the database (Bozkaya, Yazdani, & Özsoyoğlu, 1997; Keogh & Ratanamahatana, 2004; Kim et al., 2001; Sakurai, Yoshikawa, & Faloutsos, 2005; Yi et al., 1998; Y. Zhu & Shasha, 2003), they are not suitable for clustering purpose because dissimilarity matrix must be fully calculated in clustering. For example, in clustering algorithms such as the well-known Unweighted Pair-Group Method with Arithmetic Mean (UPGMA) (Sneath & Sokal, 1973), all distances must be calculated and no pruning can be done. In such cases, clustering process would benefit from a fast and accurate similarity measure (Gronau & Moran, 2007). As a result, the quadratic nature of existing methods would make this computation extremely lengthy for time-series. Using prototypes in the third step of MTC for finding similar time-series in shape which are small in size, addresses the issue. It is the motivation for using prototypes in the MTC (see Section 4.44.5.1). Moreover, finding clusters of time-series which are similar in shape are very close to ground truth and more meaningful (see Section 5.3.3.1).

- Arbitrary shape of clusters can be achieved by a sophisticated merging approach. An algorithm which can make a hierarchy at the last step is very important and intuitive. As a result, arbitrary clustering is supported in the last step in this methodology to provide arbitrary shape clusters as well.

3.2.4.2 IMTC

To address the second objective of this study, the proposed model (MTC) is extended as an interactive clustering method (IMTC). It is very useful to design a clustering model which provides the results interactively (Seo & Shneiderman, 2002). That is, a need for an interactive clustering (Grass, 1996; Zilberstein & Russell, 1995) is demanded. Interactive clustering is a clustering approach which carried out in some repetitively

steps and tries to improve the results in each iteration. Meanwhile, a user can interrupt the process of clustering, and get the generated results (best results) so far. If the results are still not satisfactory for him, then he let clustering process continues (more details in Section 4.7).

3.2.5 System Design

According to the proposed models, the representation methods, the distance measure method and algorithms are depicted in Figure 3.3.

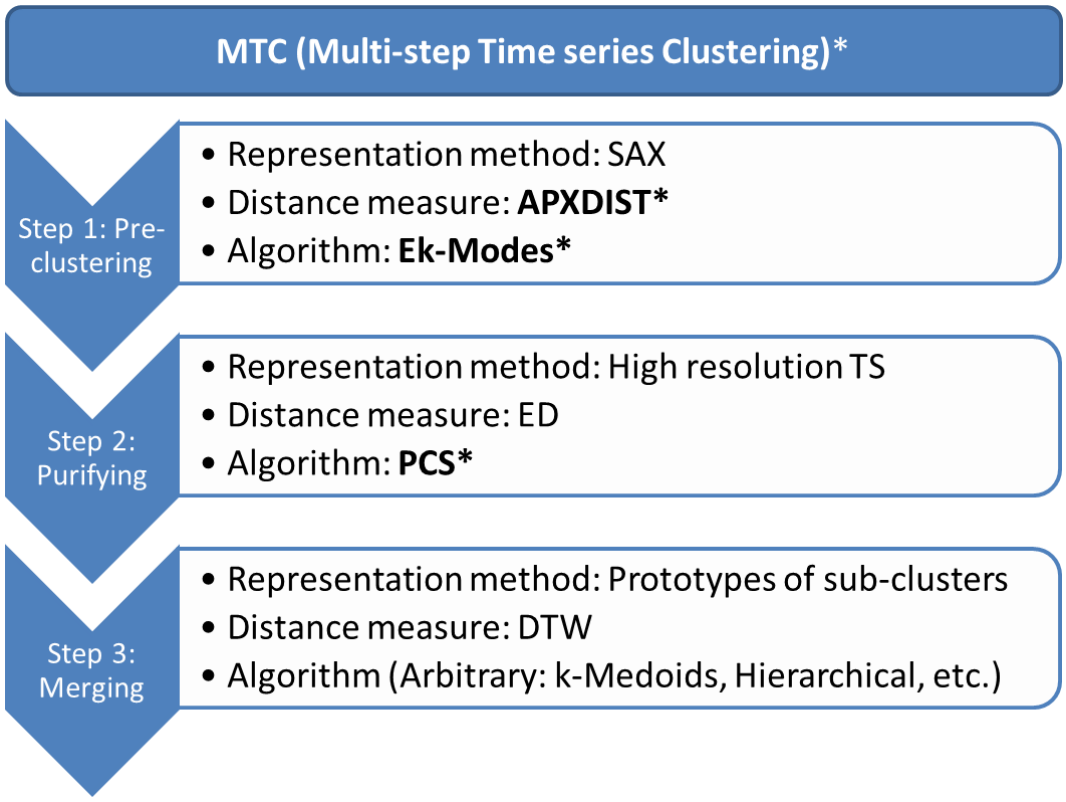


Figure 3.3: Steps of proposed model for clustering of time-series data

In addition to proposing the MTC model, the following methods are developed in each step, which are considered also as contributions of this study (highlighted by stars in Figure 3.3):

1. To develop an accurate method for calculating distance measure between time-series represented by symbolic representation (APXDIST) (see Section 4.3.3)

2. To develop an algorithm for approximate clustering of dimensionality reduced data (Ek-Mode) (see Section 4.3.4)
3. To develop a method for purifying the pre-clusters (PCS) (see Section 4.4.2)

As mentioned, MTC model is extended to be performed interactively by repeating the second and third step. Figure 3.4 shows the process of the model. Design of these two models (MTC and IMTC) is explained in the next chapter (see Chapter 4.0).

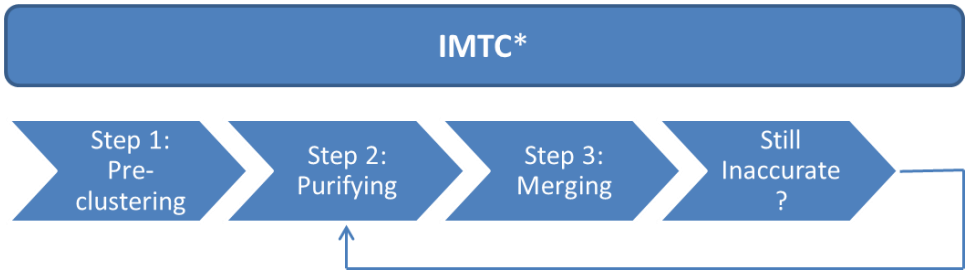


Figure 3.4: Proposed IMTC model for interactive clustering of time-series

3.2.6 Analysis of Methods

After designing the proposed methods in the MTC and IMTC models, all the steps are implemented using the MATLB software. Then, the proposed methods in each step of MTC are applied on a variety of datasets to adjust how each method improves the accuracy of the model in each step. The designed methods in each step (i.e., APXSAX, E-kModes, and PCS) are evaluated separately, compared with competitive methods and analysed. The details of experiments and results are discussed in Chapter 5.0.

3.2.7 Evaluation Method

To address the third objective of this study, the models are evaluated experimentally. Keogh & Kasetty (2003) have made an interesting research on different articles in time-series mining and conclude that the evaluation of time-series mining should follow some disciplines which are recommended as:

- The validation of algorithms should be performed on various ranges of datasets (unless the algorithm is created only for a specific set). The used dataset should be published and freely available
- Implementation bias must be avoided by careful design of the experiments
- If possible, data and algorithms should be freely provided
- New methods of similarity measures should be compared with simple and stable metrics such as Euclidean distance.

This study attempts to follow most of these suggestions in order to perform an extensive evaluation. Firstly, around 20 different datasets from different domains are utilized which are used in different articles for evaluation. These include real world datasets and syntactic datasets. Secondly, standard algorithms for clustering are used to avoid implementation bias. Moreover, the pseudocode for all the methods are provided separately. Finally, for proposed methods, the results are compared with standard and simple approaches. However, in general, evaluating of extracted clusters (patterns) is not easy in the absence of data labels (H. Zhang et al., 2006) and it is still an open problem. The definition of clusters depends on the user, the domain, and it is subjective. For example, the number of clusters, the size of clusters, definition for outliers, and definition of the similarity among the time-series in a problem are all the concepts which depend on the task at hand and should be declared subjectively. These have made the time-series clustering a big challenge in the data mining domain. However, owing to the classified data labelled by human judge or by their generator (in synthetic datasets), the result can be evaluated by using some measures. The label of human judge is not perfect in terms of clustering raw data, but in practice it captures the strengths and shortcomings of the algorithms as ground truth (see Appendix B). To evaluate MTC, the datasets are used from different domains which their labels are known. Figure 3.5 shows the process for evaluation MTC.

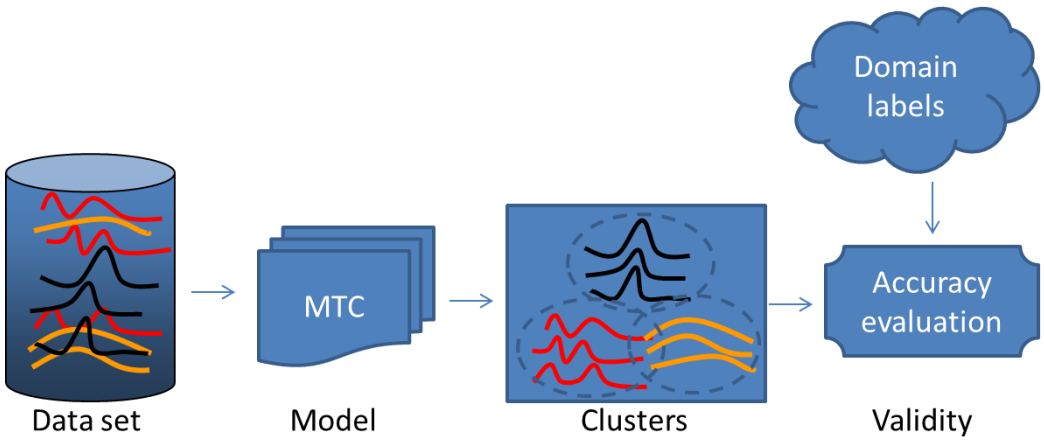


Figure 3.5: Experimental evaluation of MTC

The choice of measures in this research is based on the most common-used measures in time-series clustering in the literature review (see Section 2.7). The internal measure is based on a study of thirty measures (G. Milligan, 1981); the external measures resulted as best choices in some recent studies (Amigó et al., 2009; Brun, Sima, Hua, & Lowey, 2007; S. Lin et al., 2008; J. Wu et al., 2009).

Rand Index, Adjusted Rand Index, Entropy, Purity, Jacard, F-measure, FM, CSM, and MNI are used for the evaluation of MTC. All of these clustering evaluation criteria have values ranging from 0 to 1, where 1 corresponds to the case when ground truth and finding clusters are identical (except Entropy which is conversed and called cEntropy). Thus, here, bigger criteria values are preferred. Each of the mentioned evaluation criterion has its own benefit and there is no consensus of which criterion is better than other criteria in the data mining community. To avoid biased evaluation, the average of all measures is computed in this thesis and the conclusions are drawn based on the average value. Moreover, to report the results, the average quality of 100 runs is calculated to prevent the bias of random initialization (De Gregorio & Maria Iacus, 2010; Hirano & Tsumoto, 2007; J. Lin, Vlachos, et al., 2004; Petitjean et al., 2011; Ratanamahatana et al., 2005; Vlachos et al., 2003). For different parameter combinations, the average quality is reported as accuracy of clustering in all

experiments in this thesis. Although the focus of this study is on improving accuracy, scalability and sensitivity of the proposed model are calculated to prove its feasibility theoretically.

3.3 Chapter Summary

The methodology adopted for this research was discussed in this chapter. According to research objectives, a research methodology framework was proposed. As explained, two models are proposed and developed in this study (MTC and IMTC). These models work as multi-step clustering approach. The motivation for using the multi-step approach was discussed based on each step. The details of the proposed model and the techniques which are used in each step were explained according to the following sequence: pre-clustering, purifying and merging. Additionally, it was explained that for each step, new methods should be designed, so called APXDIST, Ek-Modes and PCS. The developed methods to achieve the objectives were mentioned here. However, the details of each step of the model are explained in the next chapter. Then, the evaluation plan for the proposed model was explained in the last step of the research methodology framework. As explained different datasets from various domains are used to evaluate the accuracy of the proposed model.

4.0 SYSTEM DESIGN

4.1 Introduction

In this chapter, the proposed model used for accurate clustering of time-series data, i.e., Multi-step Time-series Clustering (MTC), is explained and designed in detail. In Section 4.2, a general view of the model is explained which is based on a multi-step clustering. Each step is explained in the subsequent sections. At first, clusters are made in a high-level mode, considered as pre-clustering step which is explained in Section 4.3. Then, the purifying of clusters is carried out in Section 4.4 which generates sub-clusters represented by prototypes. In the third step, sub-clusters (prototypes) are merged to form final clusters in Section 4.5. Additionally, Section 4.7 explains how the whole process can be performed as an interactive algorithm (IMTC).

4.2 Overview of Proposed Model (MTC)

Time-series datasets have different characteristics, e.g., short or long, multivariate or univariate, same or various length time-series. Additionally, it may vary in different domains, for example, time-series may have big changes in the start time points and small changes after a while. Alternatively, it may vary in different datasets, for example, a dataset may include some time-series which have high frequency, whereas another may have less frequency but high noise or outliers. In order to cluster time-series data, an appropriate clustering algorithm should be adopted. Type of proper clustering algorithm for time-series data depends on the size and shape of clusters, the type of time-series, the importance of different points of time-series and some other characteristics of datasets such as size, noisiness and its number of outliers. As a result, considering all these varieties, proposing a general solution may be less effective than more specific approaches. However, all these characteristics are related to some components of time-series clustering (such as representation method, distance measure

or prototype construction) and can be solved by adopting a proper approach for that specific component (considering the domain and characteristics of its time-series), but still there need an exhaustive solution as clustering model to overcome the problem discussed in 1.3.

In this study Multi-step Time-series Clustering (MTC) is presented as a clustering model specifically for large time-series datasets in the domains which need accurate clusters (e.g., finance, healthcare). It overcomes the limitations of traditional clustering algorithms discussed in chapter 2.0.

In this approach, at first, clusters are made in a high-level mode (pre-clustering), then the accurate sub-clusters is generated (purifying and summarization), and finally, in the third step, sub-clusters are merged to form final clusters (merging).

Figure 4.1 provides an overview of the overall approach used by MTC to find the clusters in a dataset.

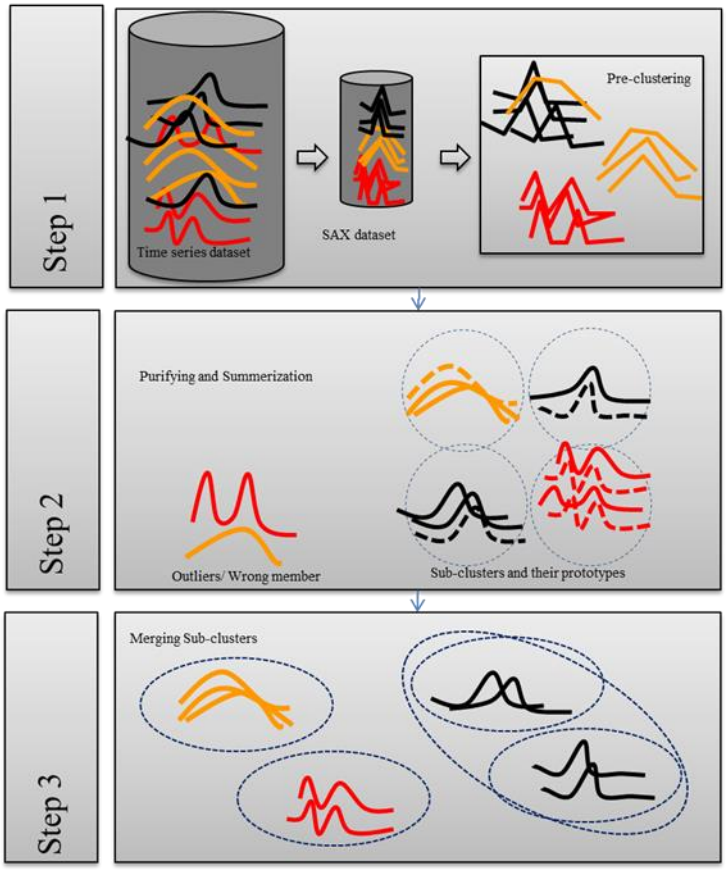


Figure 4.1: The overall view of steps of MTC

Each step of this methodology includes some activities which are mentioned in Figure 4.2.

Steps	**Activities**	**Methods**
Step 1: Pre-clustering	Activity 1.1: Pre-processing	Z-Normalization
	Activity 1.2: Reducing the dimension of time-series	SAX
	Activity 1.3: Calculating distance	APXDIST
	Activity 1.4: Perform the pre-clustering to group data	Ek-Mode
Step 2 :Purifying	Activity 2.1: Distance calculation on high resolution time-series	ED
	Activity 2.2: Purifying of pre-clusters (generate sub-clusters)	PCS
	Activity 2.3: Prototyping (Summarization)	Multi prototypes Single prototype
Step 3: Merging	Activity 3.1: Distance calculation	DTW
	Activity 3.2: Merging	Hierarchical, Partitioning, etc.
	Activity 3.3: Mapping sub-cluster objects to new clusters	-

Figure 4.2: Activities of each step of MTC

According to the above steps, activities of the MTC are explained in the following sections:

4.3 Step 1: Pre-clustering (Approximate Clustering)

In order to cluster large time-series datasets, an efficient mechanism to reduce the size of data is required. Reduction of data size can be applied to data from two aspects: reduction in number of input objects (cardinality reduction), and reducing the dimension of objects (low-resolution time-series). Reduction in input size of objects can be performed by random sampling, where, clustering is performed on a random sampling data drawn from original dataset. This approach is effective in some large datasets, and has been used in some works such as (Guha et al., 1998; Karypis et al., 1999), however sampling itself is not very straightforward (Vitter, 1985). The second approach is reduction of dimension which is also used in MTC. In the first step of the proposed algorithm (second activity), MTC focuses on reducing the dimension of time-series data. The key idea of pre-clustering is to apply clustering to the low-resolution time-series which can fit in memory rather than original (raw) time-series dataset. The obvious advantage of pre-clustering is that the whole execution time is reduced because it is run on very lower dimension of data instead of high dimensional data. Moreover, the probable noises existing in time-series are handled using time-series reduction (C. Ding et al., 2002).

The objective of this step is developing a simple partitioning scheme for speeding up the clustering. As a result, a pre-clustering is required to find approximate clusters as fast as possible with the moderate quality. However, it should be taken into account that some clusters may be missed out through clustering process on low-resolution objects which are addressed in the second step. Even though this step (pre-clustering) is a trade-off between accuracy and speed, as it is shown in the experimental result (see Section 5.3.1), considering a moderate resolution, it has tried to construct approximately good clusters.

Note that the size of dataset can be very large such that even with dimensionality reduction, it cannot fit in the memory. One solution for this case is utilizing incremental clustering. In an incremental clustering approach, clusters are updated (or expanded) incrementally. In a study (Aghabozorgi et al., 2012), the authors developed an incremental approach for clustering of time-series in large datasets which is also applicable in this step. Interested reader are referred to that work (Aghabozorgi et al., 2012).

Workflow of pre-clustering step is shown in Figure 4.3.

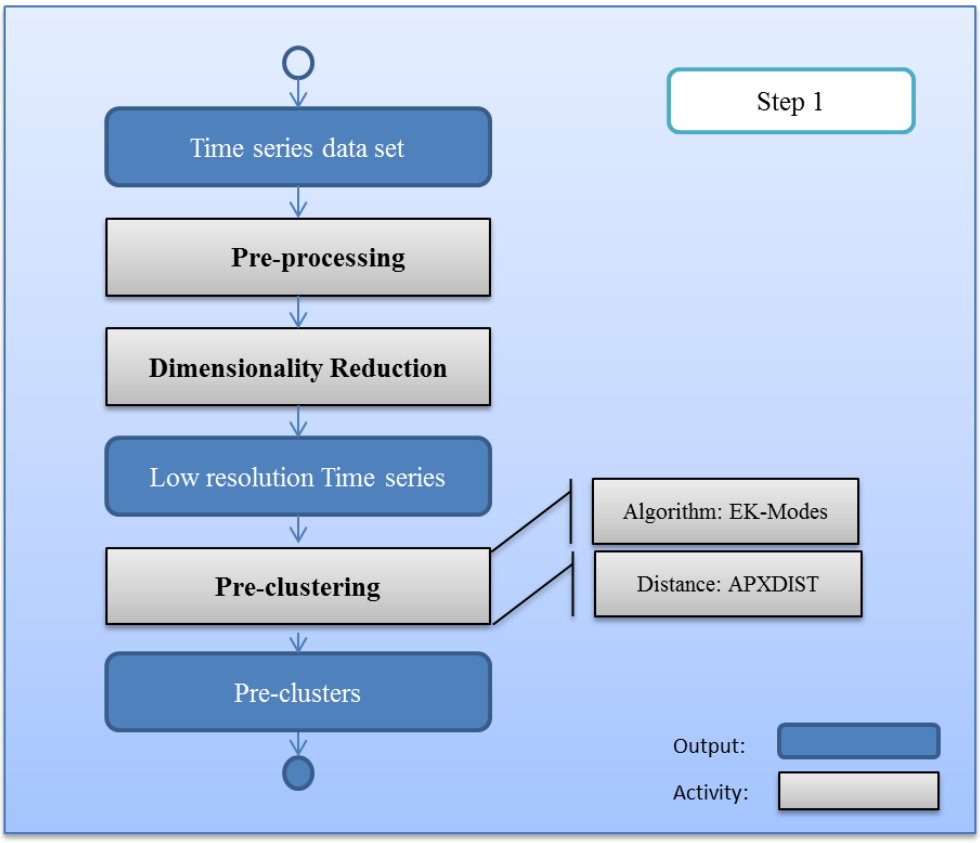


Figure 4.3: The workflow of the first step of MTC where transformed time-series are clustered.

The time execution of algorithm in the pre-clustering is directly depended on three factors: the resolution of time-series, distance calculation complexity and clustering algorithm complexity. In Section 6.3.2, these factors are discussed further. In the following sections, pre-processing, dimensionality reduction method, distance measure and pre-clustering algorithm used for pre-clustering step are explained in detail.

4.3.1 Activity1: Pre-processing

Pre-processing is necessary before attempting to match two time-series under Euclidean distance, Dynamic Time Warping or any other distance measure. It is well understood that it is meaningless to compare time-series with different offsets and amplitudes (Keogh & Kasetty, 2003). As a result, the time-series are standardized using z-score (z-Normalization) (Han & Kamber, 2011) which make time-series invariant to scale and offset. That is, each time-series is normalized in this activity to have a mean of zero and a standard deviation of one before discretizing it in next activity. It transfers time-series from absolute values to another space which is suitable for comparison. Moreover, it will decrease the sensitivity of distance measures in front of scaling.

Suppose the $F_i = \{f_1, .., f_t, .., f_T\}$ is a time-series with T data points. Z-normalization is defined as:

$$Z_{Normalization}(F_i, M_i, sd) = \frac{f_t - M_i}{sd} \quad 4.1$$

where

$$M_i = \frac{\sum_{t=1}^{T} f_t}{T} \quad 4.2$$

and

$$sd = \sqrt{\frac{\sum_{t=1}^{T} (f_t - M_i)^2}{T}} \quad 4.3$$

where M_i is an arithmetic mean of the data points f_1 through f_T, and *sd* is the standard deviation of all data points in that time-series. All time-series of each dataset are normalized in this activity. Figure 4.4 and Figure 4.5 show three raw and normalized time-series in a cluster.

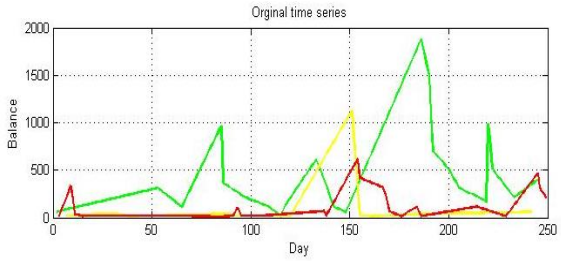


Figure 4.4: Raw time-series before normalization

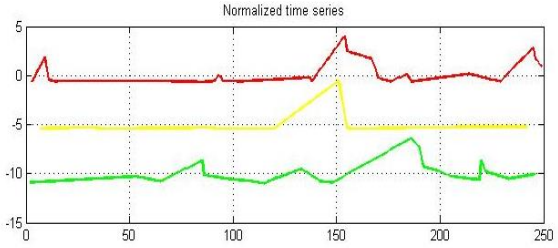


Figure 4.5: Normalized time-series

4.3.2 Activity2: Dimensionality Reduction

One can start with this question: Why dimensionality reduction should be applied on data in the first step? As it were discussed in Section 3.2.4.1 as part of the motivation for designing the multi-step approach, dimensionality reduction is important because raw time-series are high dimensional data, and distance calculation between the raw time-series is not very fast. That is, most of accurate distance measures are quadratic in the length of the time-series and it is a big challenge for raw time-series. Moreover, dimension reduction can solve the problems like noise in time-series to a high extend. It was discussed widely in the literature review (see Section 2.4).

There are many dimensionality reduction methods suggested in the literature (Section 2.4) which provide a lower-resolution data (compatible with domain). In this thesis, SAX is adopted as representation method, because of its low complexity and relatively good quality. The superiority of SAX was discussed in Section 2.4.4 in detail. However, the question is to how much reduction should be applied to dataset considering the probability of missing clusters? How much dependent is the final cluster result to the pre-clusters' quality? How much effect does missing clusters have on the final result? These questions are answered widely by experiment in Section 5.3.1.1.

To represent time-series by SAX representation, for each time-series, a reduced time-series is initialized as follows:

One can consider F_i as a time-series, where $F_i = \{f_1, .., f_t, .., f_T\}$. Then, time-series is discretized by Piecewise Aggregate Approximation as $\bar{F} = \bar{f_1}, \ldots, \bar{f_w}$. In this process, *w* is the number of PAA segments representing time-series *F*. Each segment of $\bar{F}$, i.e., $\bar{f_i}$, is a real value which is the mean of all data points in the i[th] segment of *F*, and defined as:

$$\bar{f_i} = \text{average}(f_k),\ k \in [\frac{T}{W}(i-1)+1, \frac{T}{W}i] \quad 4.4$$

Then all time-series data are transformed to $\hat{F}$ (by mapping the PAA coefficients to 'a' SAX symbols) where $\hat{F} = \hat{f_1}, \ldots, \hat{f_w}$. In this process, w is the number of PAA segments representing time-series F_i, and 'a' is alphabet size or the number of symbols (e.g., for the alphabet= {a, b, c}, a = 3). To define the alphabets in SAX, the "breakpoints" are used that will produce the equal-sized areas under Gaussian curve.

Definition 4.1: Breakpoints, "breakpoints are a sorted list of numbers $B = \beta_1, \ldots, \beta_{a-1}$ such that the area under a *N*(0,1) Gaussian curve from β_i to $\beta_{i+1} = 1/a$ (β_0 and β_a are defined as -∞ and ∞, respectively)." (J. Lin, Keogh, Lonardi, et al., 2003).

Symbols in SAX are defined based on location of the PAA values, i.e., $\bar{f_1}, \ldots, \bar{f_w}$ in each region which are defined by break points. That is, using the values of PAA and their location in the intervals made by breakpoints, each segment of $\bar{f_i}$ is coded as a symbol of $\hat{f_i}$ using the following equation:

$$\hat{f_i} = \begin{cases} A_1, & \beta_0 < \bar{f_i} \leq \beta_1 \\ A_2, & \beta_1 < \bar{f_i} \leq \beta_2 \\ A_x, & \beta_x < \bar{f_i} \leq \beta_{x+1} \\ A_w, & \beta_{w-1} < \bar{f_i} \leq \beta_w \end{cases} \quad 4.5$$

where, A_x is the x-th character of alphabet set. For example, Figure 4.6 shows a time-series converted to SAX. In this example, for alphabet size a=6, there are 5 break points B={0.97,0.43,0,-0.43,-0.97} which divide the area under Gausian curve to 6

equiprobable regions In the example, with n = 32, w = 4 and a = 6, the time-series is mapped to the word ‘efcacfreb’.

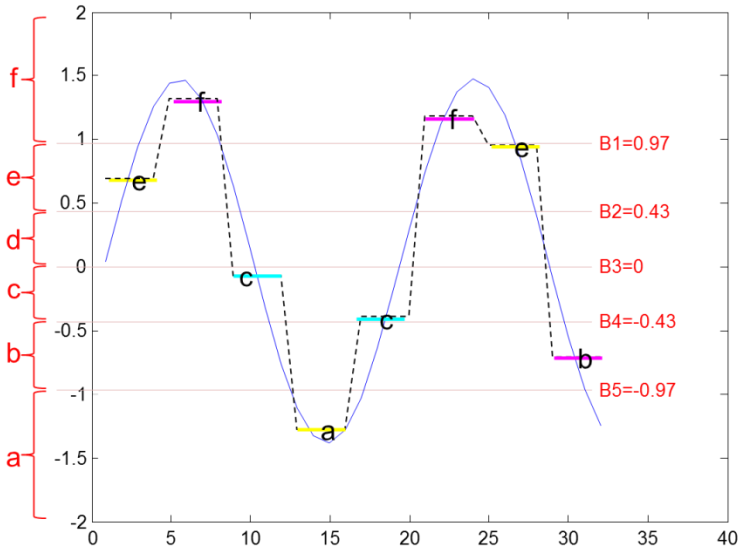


Figure 4.6: A sample of a time-series represented by SAX

Then, $\hat{F}$, the dimensionality reduced time-series, is used instead of raw time-series F. Undeniably, the dimensionality reduction has some disadvantages which should be considered in the clustering process. For example, because low-resolution time-series are used, it can lead to construction of incorrect clusters, empty clusters or missing out some certain clusters. However, the results in the experiment indicate that, first, with considering a moderate compression for SAX, generally acceptable results are obtained in the first step of MTC; second, with increasing of resolution of time-series, the accuracy of conventional clustering is not improved necessarily (see 5.1.1.3.1).

4.3.3 Activity3: Distance Calculation (APXDIST)

In order to make the pre-clusters, an appropriate distance measure compatible with SAX is desirable. In this Section a new method for distance measurement between time-series (APXDIST) is introduced. It is the answer to the first question of this study, i.e., “is there any alternative approach for increasing the accuracy of clustering of symbolised time-series?”

For dimensionality reduced time-series (using SAX), the Euclidean(Lai et al., 2010) or MINDIST (J. Lin et al., 2007) measure are used in order to calculate the similarity between two time-series. In this activity, a new distance method (APXDIST) is designed based on SAX representation which is also one of the contributions of this thesis as well. This method is explained with posing these questions: "What is the motivation for using APXDIST?"

Using SAX representation, the distance metric compatible with SAX is desirable. J. Lin et al (2007) introduced MINDIST as a compatible distance metric for SAX. The distance between two symbolized time-series is calculated by:

$$dis_{MINDIST}(\bar{F}_x, \bar{F}_y) = \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^{w} (dis(\hat{f}_i, \hat{f}_j))^2} \quad 4.6$$

where dis() function is defined as the minimum distance between symbols of represented time-series, e.g., dis(a,a) = 0, dis(a,b) = 1, dis(a,c) = 2, and etc. The dist() function is implemented using a table lookup and does not need to be calculated for each symbols, which is considered also as its outstanding feature. The dis() function is calculated by:

$$dis(\hat{f}_i, \hat{f}_j) = \begin{cases} 0, & |i-j| \leq 0 \\ \beta_{i-1} - \beta_j, & j+1 < i \\ \beta_{j-1} - \beta_i, & i < j-1 \end{cases} \quad 4.7$$

However, this distance ($MINDIST$) has been introduced to address the indexing problem in time-series domain and is not enough accurate for calculation of distance among time-series in the clustering problem, as a result, in this thesis a new approach, APXDIST is defined and depicted in Figure 4.6.

SAX is defined based on PAA (Keogh et al., 2001a; Yi & Faloutsos, 2000) and assumes normality of the resulting aggregated values (see Section 2.4.4 and 4.3.2 for more

details and definitions). Essentially, normalized subsequence have highly Gaussian distribution according to an empirical test done by (J. Lin, Keogh, Lonardi, et al., 2003). Given that the normalized time-series have highly Gaussian distribution; a distance between the symbols is introduced based on Gaussian characteristic, so called APXDIST.

In MINDIST approach, the distance between two SAX representations of a time-series requires looking up the distances between each pair of symbols. As mentioned, symbols in SAX are defined based on location of PAA coefficients in some regions or buckets. The distance between the symbols is calculated in relation to the distance between these buckets made by break points. That is, the height between the regions as illustrated in Figure 4.7. Actually, for calculating the distance between two symbols in different regions, the distance between regions should be calculated. For example, for alphabet size a=6, there are 5 break points $\beta = \{0.97, 0.43, 0, -0.43, -0.97\}$ which divide the area under Gaussian curve to 6 equiprobable regions as illustrated in Figure 4.6. Equiprobable means that the probability of a segment falling into any of the regions is approximately the same.

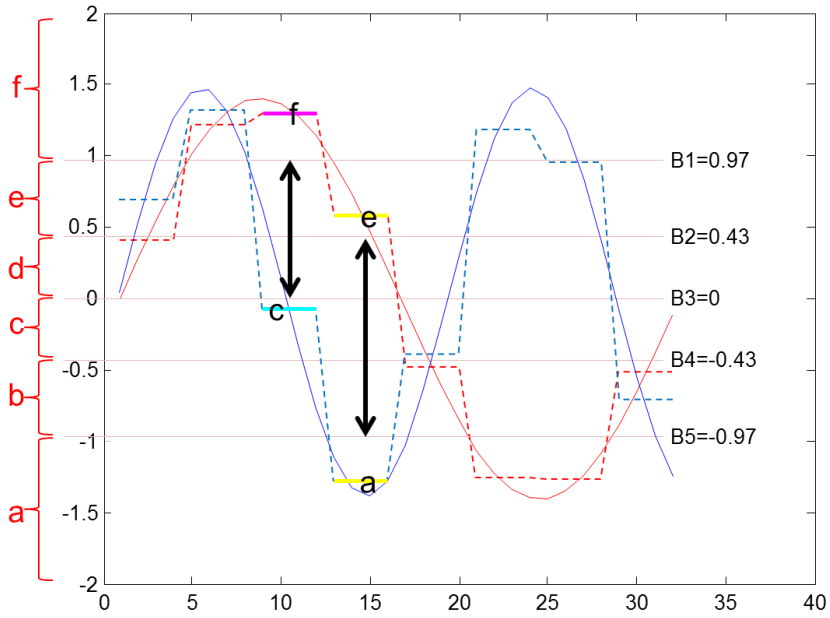


Figure 4.7: MINDIST measure for calculating similarity between the symbolized time-series using SAX

As mentioned, MINDIST is proper for indexing purpose (because of its upper bounding feature), but not accurate for clustering, because based on its definition, it considered the distance of neighbour symbols as zero, and ignores the maxima and minima points of time-series. As a result, a more precise distance measure is defined, to address the shortage of MINDIST. The key idea behind the APXDIST is that a more precise distance measure can be defined to calculate the distance between the regions than their minimum height, i.e., the distance between regions can be calculated as distance between indicators of the regions. It is to reduce the probability to miss some important points in time-series in upper or lower regions (e.g., the maximal and minimal points indicated by symbols ‘f’ or ‘a’ in Figure 4.7), and dissimilar adjacent symbols which are considered as similar symbols (e.g., the dis(a,b)=0 in MINDIST, see Appendix C). For each region, an indicator is defined in such a way that the closeness of PAA coefficients (in the region) to the indicator is the highest in that region. For defining the indicator, the distribution of PAA coefficients in each area is considered. For example, distribution of a sample area for a symbol (e.g., ‘e’) is considered as depicted in Figure 4.8.

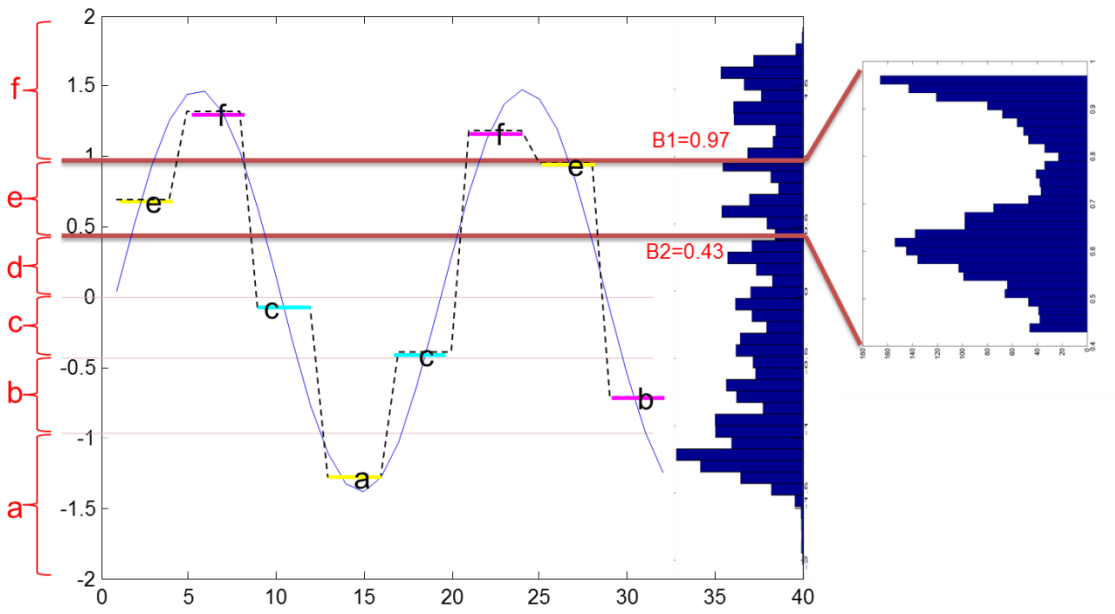


Figure 4.8: distribution of a sample area for a symbol (e.g., 'e')

Then, arithmetic mean of each area (minimum and maximum) is defined as indicator of the area as the best estimator of the regions as:

$$Ind_i = \frac{\beta_{i-1} + \beta_i}{2} \qquad 4.8$$

where β_0 is global minimum and β_a is global maximum. For example, six indicators are defined for a=6, i.e., Ind={(Max+0.97)/2, 0.70, 0.21, -0.21, -070, (Min+0.97)/2}. Here, Max (and Min) indicated the global maximal (and minimal) of time-series in a dataset. Indicators are used in APXDIST to calculate the distance between to symbols in two different regions associated with the alphabetic symbols. For example, mean line on the area between cut lines of 0.97 and 0.43, which indicated the alphabet 'e', is 0.70. Figure 4.9 illustrates a visual intuition of the measure. The black line shows the MINDIST between two symbols, while green line indicates the APXDIST distance between the same symbols. Accordingly, the indicator of region related to 'a', that is the region below the cut-line -0.97, is the mean line (Min-0.97)/2, where Min means the minimum value of PAA coefficients for a dataset. As a result, the distance between the symbols 'e' and 'a' is calculated as distance of two mean lines, that is 0.70-(Min-0.97)/2.

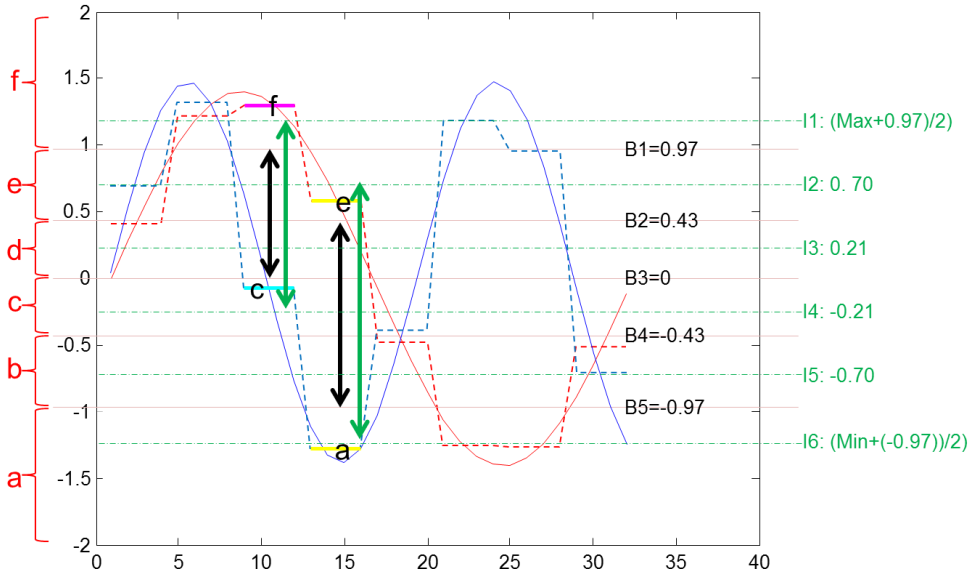


Figure 4.9: For calculation of distance between the time-series in Figure 4.6 and another time-series, for each area an indicator is defined.

This approach for calculation distance between the regions (symbols) results in a tighter distance measure. Accordingly, the distance between two SAX representations of two time-series requires looking up the distances between each pair of symbols. In APXDIST the distance between two pair is approximate distance between two indicators, which are also indicated by a lookup table which make the calculation very fast because it is pre-calculated. Then, same as the MINDIST procedure, squaring them, summing them, taking the square root and finally multiplying by the square root of the compression rate ($\frac{n}{w}$). Based on this definition, the APXDIST between each pairs of symbolized time-series is defined as following:

$$dis_{APXDIST}(\bar{F}_x, \bar{F}_y) = \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^{w} (dis(\mathrm{Ind}_i, \mathrm{Ind}_j))^2} \qquad 4.9$$

where ind_i is the indicator of i_{th} region and the dis() function is calculated by

$$dis(\mathrm{Ind}_i, \mathrm{Ind}_j) = \begin{cases} 0, & i = j \\ \left| \frac{-\beta_j + \beta_{i-1} + \beta_i - \beta_{j-1}}{2} \right|, & else \end{cases} \qquad 4.10$$

and $dis(\mathrm{Ind}_i, \mathrm{Ind}_j)$ is computed by a lookup table. For example, for alphabet of cardinality of 6, i.e., a=6, the lookup table is illustrated in Table 4.1. In this table, the distance between two symbols can be read off by examining the corresponding row and column.

Table 4.1: A lookup table used by the APXDIST function. This table is a sample for an alphabet size of 6 with the min=-1.2 and max=1.2.

APXDIST	Ind_a	Ind_b	Ind_c	Ind_d	Ind_e	Ind_f
Ind_a	0	0.38	0.87	1.08	1.78	2.16
Ind_b	0.38	0	0.49	0.91	1.40	1.78
Ind_c	0.87	0.49	0	0.42	0.91	1.08
Ind_d	1.08	0.91	0.42	0	0.49	0.87
Ind_e	1.78	1.40	0.91	0.49	0	0.38
Ind_f	2.16	1.78	1.08	0.87	0.38	0

4.3.4 Activity4: Pre-clustering (Ek-Modes)

In order to make the pre-clusters, a clustering method compatible with SAX is desirable. Here, a new clustering algorithm (Ek-Modes) is developed for pre-clustering of dimensionality reduced time-series which is explained in the follows.

For the approximate clustering of time-series in MTC, any partitioning approaches such as k-Means or k-Medoids, or even hierarchical approaches (if dataset is not that large) such as agglomerative or divisive approaches can be used. However, partitioning clustering is preferred for MTC. “Why partitioning clustering is better choice for clustering of dimensionality reduced data?” Motivation for using partitioning clustering is its simplicity, high speed (especially in large datasets) (Huang, 1998) and its rather good quality (it is shown in Section 5.3.1.3). Moreover, a study (C. Ding et al., 2002) shows that choosing the centroids on a low dimension approximation of data increases the quality in the partitioning clustering. Hence, performing partitioning clustering on approximated data leads to better quality while has its ability in fast clustering. Furthermore, using a comparison, the partitioning clustering algorithm using SAX

produces better results than using raw data (C. Ding et al., 2002). Now the question is: "Which partitioning clustering algorithm is better?"

The Mode, Mean, Median and Medoid are different popular methods of determining representative for clusters in partitioning clustering. With respect to these prototypes, k-Means, k-Median and k-Medoids are introduced in the literature. In many works, k-Medoids is used (instead of k-Means) due to its robustness to outliers (Andreopoulos et al., 2009) which are unavoidable in time-series dataset. In k-Medoids, the representative (prototype) of a cluster is one of the time-series within the cluster which has the maximum similarity to others. However, in this activity of MTC, for the first time, an extended k-Modes (Ek-Modes) is introduced and used in order to divide N time-series into k partitions which construct higher quality of clusters. The following are the reasons why k-Modes is a better choice in set of partitioning algorithms:

1) Because the SAX data is categorical data in each segment, and k-Modes work finely with categorical data (Huang, 1997, 1998).
2) Efficiency of k-Modes is high, especially for categorical data (Huang, 1998) and meet the time-series in the first step which are represented as symbolized data.
3) Because the centroid of clusters is made based on the modes (not the mean), it is robust in front of outlier time-series.

It is the first time that k-Modes are used as a solution for symbolized time-series clustering (to the best of author knowledge). k-Modes algorithm (Huang, 1997, 1998) is based on k-Means family algorithm, and is used for clustering categorical data in different works (Andreopoulos, An, & Wang, 2005; Manganaro, Paratore, Alessi, Coffa, & Cavallaro, 2005).

In Ek-Modes, at first, k cluster centers C_k^0 of same dimensionality as the time-series represented by SAX are chosen. They are initialized either randomly, or in such a

manner that they are not outlier time-series. Iterative starts with the set of initial centers, then, the distance between rest of symbolized time-series and the chosen centers C_k^0 are calculated. However, the distance measure is different here from conventional k-Modes. That is, instead of total mismatched of corresponding attributes of two objects, here, a distance based on categorical data, APXDIST (see Section 4.3.3) is used to create dissimilarity matrix. Let the inverse of $dis_{APXDIST}(\hat{F}_i, \hat{F}_j)$ indicates the similarity between two time-series. Then, the objective is to find k partitions of approximated time-series as compact (in terms of the similar time-series in the clusters) and separated (in terms of the distance between the clusters) as possible (i.e., minimize the square error). Accordingly, the lowest variation of within-cluster is found as cost function which is calculated by:

$$SSE = \sum_{i=1}^{k} \sum_{\hat{F}_i \in \hat{C}_i} dis_{APXDIST}(\hat{F}_i - R_i) \quad 4.11$$

Where $\hat{F}_i$ is an approximated time-series of pre-cluster $\hat{C}_i$ and R_i is the prototype (representative) of the i_{th} cluster.

Subsequently, the prototype of each cluster is calculated (updated). Here, a prototype is the mode of each cluster and is considered as the cluster center, representing all dimensionality reduced time-series within that cluster, that is:

$$R_k^{i+1} = \text{Mode}(C_k) \quad 4.12$$

Assume $\hat{C}_k = \{\hat{F}_1, \ldots, \hat{F}_n\}$ is a set of time-series data of length *n* within a cluster where $\hat{F}_i = \{\hat{f}_1, \ldots, \hat{f}_n\}$. Let the domain of $\hat{F}_i$ be the alphabet W defined in SAX by a categorical attributes $w_1, w_2, .., w_a$. Then the prototype of the cluster is the mode of $\hat{C}_k$, and is defined as a vector $\hat{R}_k = \{\hat{r}_1, .., \hat{r}_i, .., \hat{r}_n\}$, where $\hat{r}_i$ is the most frequently occurring value in W:

$$\hat{r}_i = w_j \mid \hat{f}_i = w_j \; and \; |w_j| > |w_x| \qquad \forall w_x \in W, \qquad \forall \hat{F}_i \in \hat{C}_k \qquad 4.13$$

For example, considering three time-series represented by SAX in the cluster $\hat{C}_k$, $\hat{F}_1$: abbcc, $\hat{F}_2$: acccd, $\hat{F}_3$:bbbcc , the "mode" of cluster is defined as: $\hat{R}_k$=abbcc

Using this technique, the prototype of a cluster time-series data is constructed. Then, iterations are continued, until convergence, i.e., until all cluster centers get stable. In general, after each iteration, the quality of the clusters and the modes themselves will essentially be improved.

Using the modes, a cluster with strong intra-similarity is obtained when the objects are categorical. It results in an efficient clustering of large categorical datasets. The efficiency of Ek-Modes (as an special case of k-Modes) is same as k-Means due to use of similar process in k-Means and k-Modes (Huang, 1998). Moreover, calculating the mode, leads to robustness of the proposed approach to outliers, since the mean point can get easily influenced by outliers (Andreopoulos et al., 2009).

The pseudo code for pre-clustering is given in Figure 4.10. Two input parameters are taken in this algorithm. The k is the first parameter which determines the final number of clusters desired, so called, the natural clusters in the dataset. It is assume that the user has a good guess for *k*, like most other methods. The second parameter is the dataset which is dimensionality reduced data.

Input: K= number of clusters

$\widehat{D}$: A time-series dataset represented by SAX

Output: C: Approximated clusters

Method: Ek-Modes (K, $\widehat{D}$)

1. If k =0, decide on a value for K.

2. Initialize K cluster centers R_k^0 of same dimensionality as symbolized time-series by randomizing data objects

3. iteration i=0

4.Dis=calculate distance (using APXDIST) between each particular symbolized $\hat{F}_x$ and the centers R_k^0

5. Assign each $\hat{F}_x$ to the cluster with the nearest center R_k^0.

6. Set new cluster centers R_k^{i+1} to the center of each cluster:

R_k^{i+1}=Mode(C_k)

7. If none of the N objects changes membership, the clustering is complete. Otherwise, repeat steps 4 to 7.

8.return C

Figure 4.10: Pseudo code for pre-clustering by Ek-Modes

The superiority of using Ek-Modes to other partitioning algorithms is shown experimentally in Section 5.3.1.3.

4.4 Step 2: Purifying and Summarization

The main objectives of this step of MTC are refining of formed clusters and summarization. The first target is to refine (purify) pre-clusters, and improve their quality. The second part is to provide some prototypes in order to reduce the complexity of algorithm. Figure 4.11 shows the activities of this step.

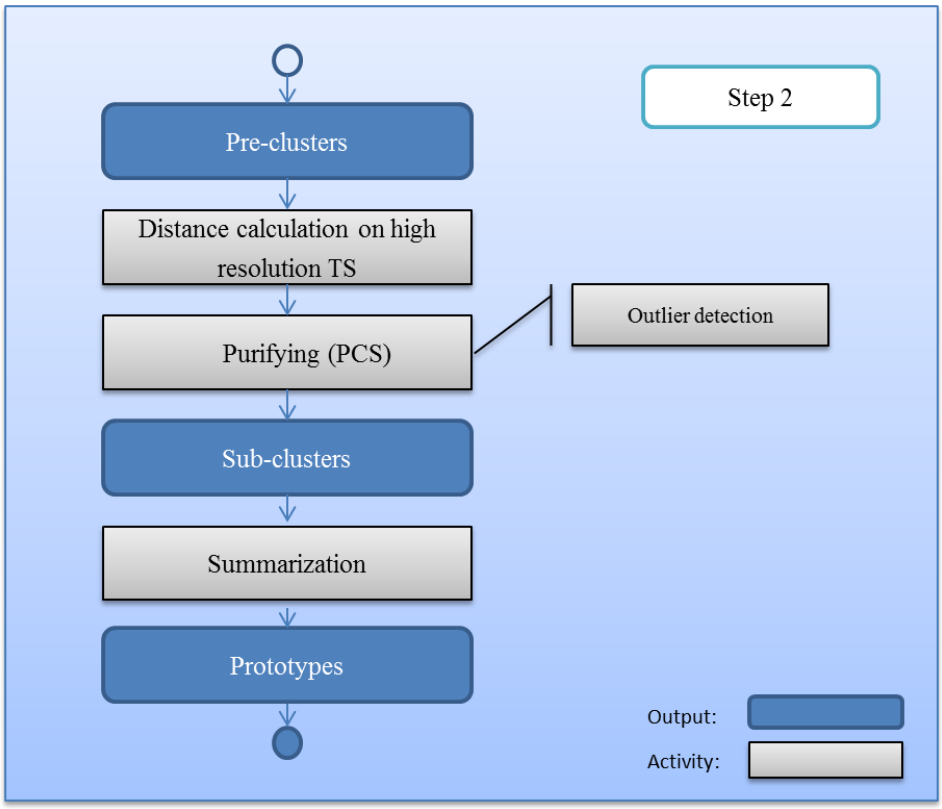


Figure 4.11: Activities of second step of MTC

Purifying: In the first activity, it attempts to increase the quality of each formed cluster (entered from the first step or the third step) by replacing its low resolution time-series with higher resolution time-series and then, by splitting the clusters. Concept of Cluster Affinity Search Technique (CAST) (Ben-Dor, Shamir, & Yakhini, 1999) is used and extended in this thesis to refine formed clusters. In this activity, a pre-cluster (approximated cluster) entered from previous step (or from third step in IMTC model), is broken down into sub-clusters based on similarity in time. Euclidean distance is used in the second step, considering that Euclidean distance is not able to reveal dissimilar time-series in shape (e.g., those which have shift) in a cluster, but at least it separates them as a new cluster if they are very far from others (mis-clustering in the pre-clustering). These separated clusters will be handled in the third step by employing an elastic distance method and merging which fall time-series into their correct clusters. The key point of third step is capability of utilizing DTW distance (to find similar time-series in shape). DTW can find the clusters based on similarity in shape, and its elastic

feature helps it to overcome the shortages of one-to-one matching and lack of shift support in approaches such as of Euclidean distance (see Section 2.5.3).

Summarization: The second activity is defining representatives for sub-clusters based on the context of utility of clusters. The motivation for defining representative is reduction of complexity in the third step. Many clustering algorithms, such as Hierarchical have a complexity of higher than O(n^2) which their utilization on large datasets is not practical. This problem is more crucial in time-series datasets because they are essentially high-dimensional, and thus, is more challenging. In the proposed model (MTC) also there is the same issue where it is merging well separated clusters (in the third step), because they consist high resolution time-series (can be raw time-series), and consequently results in high complexity in computation. To address this issue, considering that clusters formed in this step, have a high similarity in time; instead of using all time-series located in sub-clusters, their prototypes are used. By finding and storing the representative/representatives for each sub-cluster in the second step, the input size of data is reduced without approximately missing the quality, and with a guarantee that it fits in the main-memory. In the following sub-sections main activities and used methods of this step are described in detail:

4.4.1 Activity1: Calculate Similarity

This activity is very simple and straight forward. Considering each pre-cluster, its members (which are time-series represented by SAX) are replaced with the corresponding time-series, but with higher resolution. It is clear that higher resolutions will result in higher quality in the final results. However, the raw time-series may not be a good choice as highest resolution for datasets which are very noisy because it affects quality of purifying of the pre-clusters. As a result, in noisy datasets, it is probable to use a lower compression in compare to first step.

In the first step, only the approximate similarity in time was calculated for each pair. Here, a more accurate similarity measure on higher resolution of time-series is calculated. For this step, Euclidean distance is used as similarity measure to calculate the distance between time-series data in each pre-cluster. The distance measure calculated by Euclidean distance is more accurate than APXDIST, because its calculation is based on higher resolution of data. Moreover, ED takes all data points of time-series into account, and compares each pair of data points in time, that is, similarity-in-time of time-series is computed. Figure 4.12 depicts the intuition behind using ED in the second step.

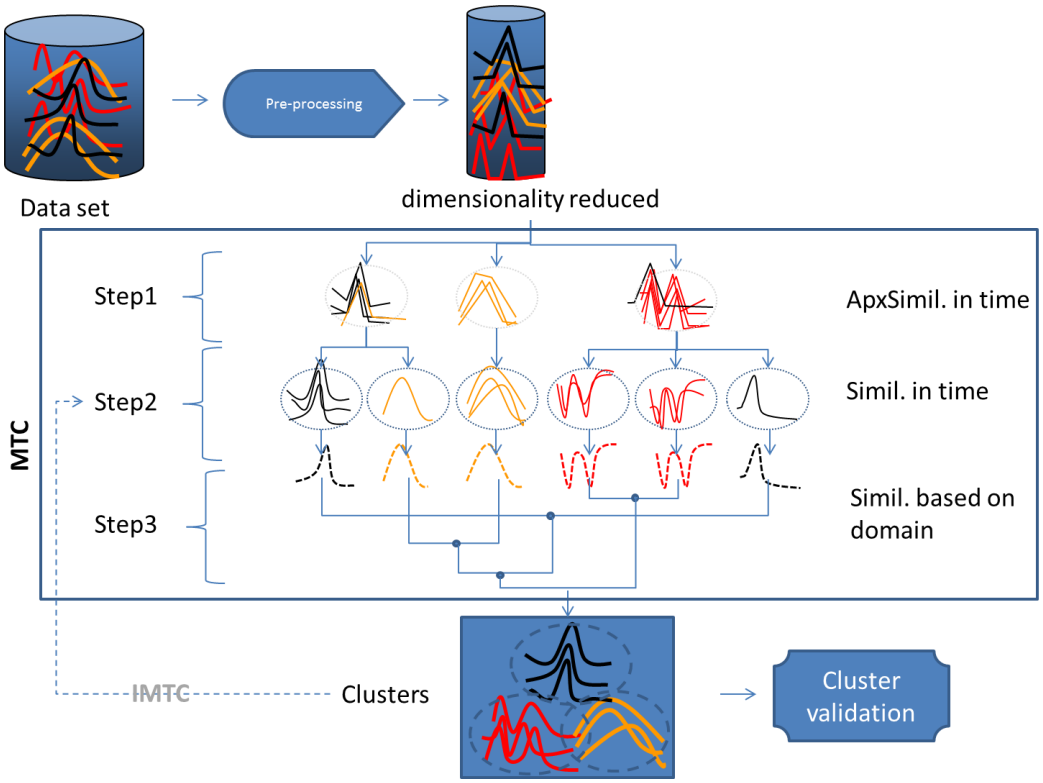


Figure 4.12: Similarity calculation in different steps of MTC

4.4.2 Activity2: Purifying of Clusters (PCS)

The pre-clusters constructed in the first step are refined here. The dilemma is: given a cluster, it should be decomposed into separated sub-clusters as pure as possible.

Definition 4.2: Sub-cluster, a sub-cluster SC_{ij} is a set of individual time-series that are close to each other (similar in time), created by splitting of pre-clusters PC_i, and will be represented as a single prototype (or more than a prototype).

Here, the purity of a cluster (or a pure cluster) is defined as follows:

Definition 4.3: Pure cluster, a cluster is pure if all its members are of the members of natural cluster (ground truth).

It means that the sub-clusters (made from a pre-cluster) are desirable such that most members are members of a natural cluster (class). That is, because the pre-clusters are generated approximately, not precisely, they are often mixed with time-series from different classes. Therefore, they should be recognized and separated by searching in the pre-clusters. As a result, pre-clusters are broken into so called pure sub-clusters. Of course assigning each object to a separate cluster (singleton cluster) provides the highest purity. However, the best answer is the purest clusters and smallest number of clusters.

For illustrative purposes, a simple diagram in 2-dimensional space is used in order to describe the intuition behind the process of splitting approximated pre-clusters (see Figure 4.13).

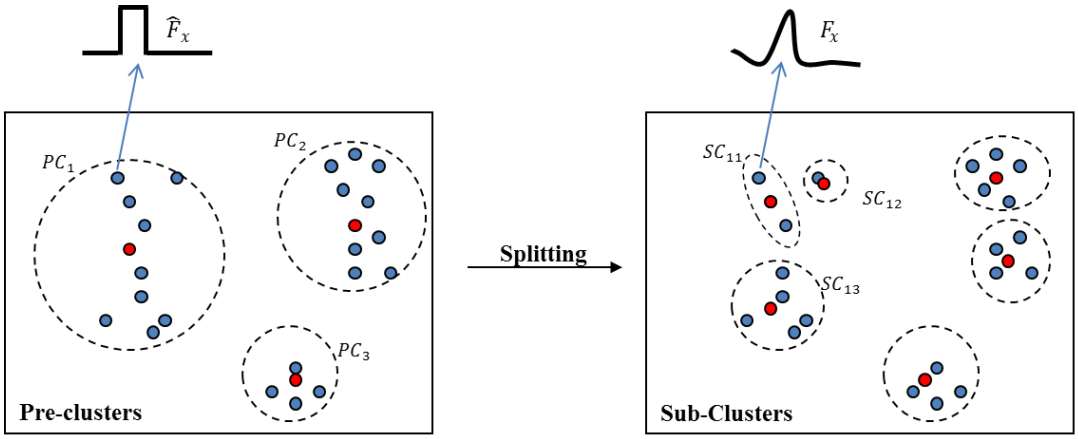


Figure 4.13: A 2-dimensional Pre-Clusters and Sub-clustering

Each time-series in a specific pre-cluster PC_i is considered as an object in an n dimensional space. Then a similarity measure between objects can be calculated and stored in an m-by-m similarity matrix, $M_{m\times m}$, where M_{ij} is the distance (similarity) measure between time-series i and time-series j. Then, a developed algorithm based on affinity search, Pure Cluster Search (PCS), is carried out on the similarity matrix to find whether cluster members are scattered solidly (see Section 4.4.2.2). Then, for pre-cluster PC_i which its members are not scattered solidly, it divides the members into sub-clusters. At that point, for each sub-cluster, a time-series is selected as prototype for further clustering.

In the next section, cluster affinity concept is explained. Then, the process of selecting prototype is given in Section 4.4.3. The experimental results in 5.3.2 verify that this process can reduce the size of data in some datasets by approximate 70% per cent, but not reduce the purity greatly.

4.4.2.1 Motivation for Using Cluster Affinity Concept

The concept of cluster affinity is borrowed from Cluster Affinity Search Technique (CAST) which used to find close objects. It is used in MTC because its output is of discrete clusters which are separated without predetermining the number of clusters. In contrast to many algorithms where the number of clusters must be predefined in advance, the mechanism used in CAST algorithm can find clusters dynamically, and deals with outliers effectively (Jiang & Tang, 2004). Hence, the concept of cluster affinity in CAST is extended here to develop a new algorithm, PCS, to split the pre-clusters to more pure clusters.

Before explaining the method of Pure Cluster Search (PCS) used for splitting of pre-clusters, at first CAST algorithm is explained briefly.

Cluster affinity search technique (CAST) (Ben-Dor et al., 1999) is based on graph theoretic method and relies on the concept of a clique graph. CAST clustering algorithm has a high performance in gene expression (Ben-Dor et al., 1999) clustering and also used for time-series mining (Lai et al., 2010). In view of the clique graph, each object is considered as a vertex in the graph. Similarity matrix which is based on the distance of objects makes indirect edges of graph. CAST makes clusters (sub-graphs) such that every object would be completely similar to every other object in the sub-graph and completely dissimilar to every object not in the sub-graph.

Although number of clusters produced by CAST does not have to be predetermined in advance, it requires a connectivity threshold which is defined by an input parameter, so-called, affinity threshold. This parameter is defined by the user and indirectly controls the size and number of clusters by determining the minimum required similarity between an object and a cluster, in order to assign an object to a cluster. CAST algorithm is sensitive to this parameter which is considered as a shortage for an algorithm, because the size and quantity of the clusters produced by the algorithm is directly affected by this parameter.

4.4.2.2 Pure Cluster Search (PCS)

PCS creates sub-clusters (from pre-clusters) sequentially with a dynamic affinity threshold. In this process, each sub-cluster is constructed with a time-series and gradually is completed by new time-series added to sub-cluster based on the average similarity (affinity) between unassigned time-series (in pre-cluster) and the current sub-cluster members. Defining a specific threshold value, the cluster accepts high affinity time-series. That is, an affinity threshold, α, is specified to determine what is considered significantly similar. This parameter controls the number and sizes of the produced clusters. After forming a sub-cluster, PCS deletes the low affinity objects from sub-

clusters. These adding and removing to a sub-clusters is performed consecutively until no more changes occur in the sub-cluster.

Definition 4.4: Cluster affinity, the affinity of a time-series F_x to a sub-cluster SC is defined as follows (4.14):

$$\mathrm{a}(F_x) = \frac{\sum_{\mathrm{y}\in \mathrm{c}} Sim(F_x, F_y)}{|\mathrm{SC}|} \quad 4.14$$

where, *Sim* is the similarity between time-series F_x and F_y, and |SC| is the number of time-series that exist in the sub-cluster. As mentioned, an affinity threshold, α, of a cluster C is defined to create sub-clusters of high affinity time-series. It is defined dynamically which is very important. One of the positive points of a data mining algorithm is its low number of parameters or preferably parameter-free. An algorithm which is parameter-free would limit our ability to impose our prejudices, expectations, and presumptions on the problem at hand, and would let the data itself speak to us. PCS is proposed as an algorithm which works without predetermining parameters. In this approach, the affinity threshold is calculated based on the affinity of each time-series in pre-cluster. The affinity threshold, α, is calculated dynamically based on the remaining time-series in pre-cluster, i.e., unassigned time-series in pre-cluster $\mathrm{PC_U}$, before constructing each new sub-cluster $\mathrm{SC_{new}}$ as:

$$\alpha = \frac{\sum_{x,\mathrm{y}\,\in PC_U, Sim(F_x,F_y)\geq M}(M_{xy} - M)}{|PC_U|} + \mu \quad 4.15$$

where

$$M = \frac{\sum M_{xy}}{|\mathrm{PC}|} \quad 4.16$$

is the mean of similarities of each time-series to other time-series in pre-cluster (M is initialized one time in the start of algorithm), and |PC| is the number of all time-series in the pre-cluster. The time-series that exist in a pre-cluster have not a unify affinity to

each other, because pre-clusters are made based on an approximated distance measure (e.g., APXDIST) and a low resolution time-series (e.g., SAX). The Equation 4.15 calculates a threshold based on the within variance of time-series in the cluster dynamically before creating each new sub-cluster. This value is used to distinguish the time-series which have not clustered properly (or outliers) by putting them into new sub-clusters. As a result, the output of this algorithm is some refined clusters which are constructed by breaking down of the pre-cluster. Figure 4.14 shows the pseudo code related to PCS.

Input: Sim : an n-by-n similarity matrix

Output: C: a collection of sub-clusters

Method: PCS(Sim)

Initializations:

1. $SC_{new} \leftarrow \emptyset$ /* The constructing sub-cluster */
2. $PC_U \leftarrow \{F_i, \ldots, F_n\}$ /* Elements inside the pre-cluster */
3. M ←the average similarity between time-series in the pre-cluster
4. while ($PC_U \cup SC_{new} \neq \emptyset$) do
5. for all F_x, F_y in PC_U
6. If $Sim(x, y) \geq M$
7. $$\alpha = M + \frac{\sum Sim(F_x, F_y) - M}{|PC_U|}$$
8. end If
9. end For
10. let F_U be an element with maximal affinity in PC_U.
11. if (a(F_U) >= α) /* F_U is of high affinity */
12. $SC_{new} \leftarrow SC_{new} \cup \{F_U\}$ /* Insert F_U into SC_{new} */
13. $PC_U \leftarrow PC_U \setminus \{F_U\}$ /* Remove F_U from PC_U */
14. for all x in $PC_U \cup C_{new}$ and F_U in SC_{new} do

15. a(F_x) = $\sum$ M(F_x, F_U)/|SC_{new}| /* Update the affinity */
16. end
17. else /* No high affinity elements outside C_{new} */
18. Let F_v be a vertex with minimal affinity in C_{new}.
19. if (a(F_v) < α) /* F_v is of low affinity */
20. $SC_{open} \leftarrow SC_{new} \setminus \{F_v\}$ /* Remove F_v from SC_{new} */
21. $PC_U \leftarrow PC_U \cup \{F_v\}$ /* Insert F_v into PC_U */
22. For all F_x in $PC_U \cup C_{new}$ and F_U in SC_{new} do
23. a(F_x) = $\sum$ M(F_x, F_U)/|SC_{new}| /* Update the affinity */
24. end
25. else /* SC_{new} is clean */
26. C ← C ∪ SC_{new} /* Close the cluster */
27. SC_{new} ← Ø /* Start a new cluster */
28. a(.) ← 0 /* Reset affinity */
29. end
30. end
31. end
32. return the collection of sub-clusters, C.

Figure 4.14: using PCS for purifying of pre-clusters

4.4.3 Activity3: Summarization (Making Prototypes)

One of the strengths of MTC is its flexibility in making different cluster shapes by choosing arbitrary algorithm for merging. That is, given the sub-clusters and defining a similarity measure, many distance based algorithms such as partitioning, hierarchical, or density based clustering can be used for merging the sub-clusters in the third step. However, instead of applying the clustering algorithm to the entire data, only the prototype/prototypes of each sub-cluster are participated in merging process which reduces the complexity of process to a high extend. Moreover, prototypes can increase

the accuracy to some extent. That is, highly unintuitive results may be garnered because some distance measures are very sensitive to some "distortions" in the data. However, because each prototype is made by averaging of some time-series, it decreases the effect of distortions or outliers in time-series. Nevertheless, one of the controversial problems is defining the prototypes (centroid or representative) of sub-clusters. The objective of this activity is finding the shape-based time-series average which is an essential subroutine in MTC. The quality of final clusters is highly dependent on the quality of averaging. Many attempts for defining an effective method for creating the prototypes for time-series clusters have been discussed in the literature review (see Section 2.6). In this study, single representative and multi-representative solutions are used for prototyping of sub-clusters. Choosing either single or multi representation, depends on the clustering scheme which is used in the third step and the required precision of results as well.

Solution 1) Single-representative: ED is used to find the average shape of two time-series. It is similar to finding centroids in k-Means algorithm. Because the number of time-series in each sub-cluster is not very big and it is guaranteed that they are similar in time, therefore, averaging method is effective for making the prototypes.

Solution 2) Multi-representative: In order to define the multi-representative, the centroid of sub-clusters is used together with some other time-series of sub-clusters. They create a set of prototypes which effectively represents the whole sub-cluster's members. This approach is used in the case that a clustering algorithm for generating arbitrary shape clusters is used for merging the sub-clusters.

4.4.3.1 Single-representative

Single representative is defined for each sub-cluster to minimize a known criterion function, i.e., the Sum of Squared Error (SSE). Given a set of n time-series in a sub-cluster CS_z, the time-series are represented by a time-series $R_z = \{r_1, \dots, r_x, .., r_T\}$

$$r_x = \frac{\sum_{i=1}^{n} f_{ix}}{n} \tag{4.17}$$

where $F_i = \{f_{i1}, \dots, f_{ix}, .., f_{iT}\}$ is a time-series in CS_j. Figure 4.15 shows the representative of sub-cluster made by SSE technique. The distance between all TS in this sub-cluster to the representative is minimum distance.

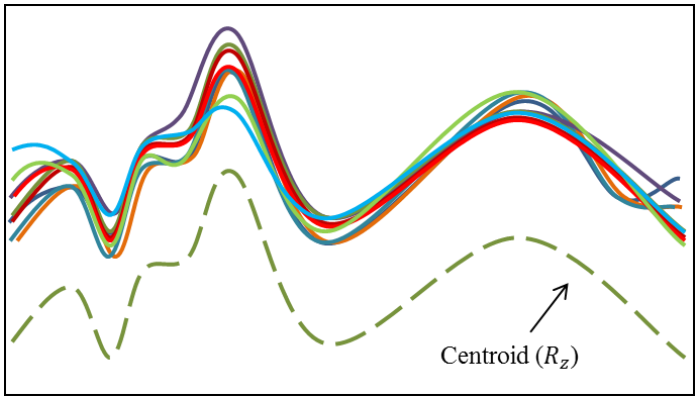


Figure 4.15: single-representative of a time-series sub-cluster

4.4.3.2 Multiple-representative

In order to determine the representatives of the sub-clusters, the concept of defining representative in CURE algorithm (Guha et al., 1998) is employed. For this activity, a multi-representative is used instead of one representative to merge clusters. The multi-representative approach is chosen because it can better represent the shape of sub-cluster, especially when the sub-clusters are big or the target merging scheme is an algorithm which generates arbitrary shape clusters. For example in Figure 4.16, the same sub-cluster in Figure 4.15 is depicted with two representatives.

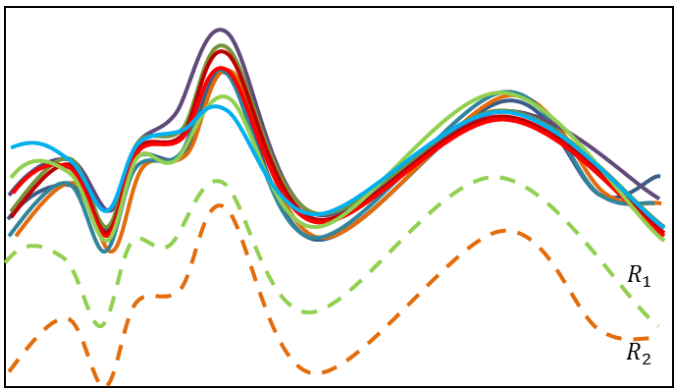


Figure 4.16: Multi-representative of a time-series sub-cluster

In this approach, a constant number of time-series between centroid and all extreme time-series are chosen as representative of a cluster. The representatives are well-scattered time-series data in a cluster which are shrunk toward centroids. Defined representatives should capture the shape and extent of the sub-clusters. The main reason for shrinking of representatives is alleviating the effect of outliers. The outliers generally have greater distance from the centroid of clusters and less distance to farther time-series of a cluster. Shrinking representatives toward centroid with fraction δ, will decrease the effect of outlier time-series in merging step as illustrated in Figure 4.17.

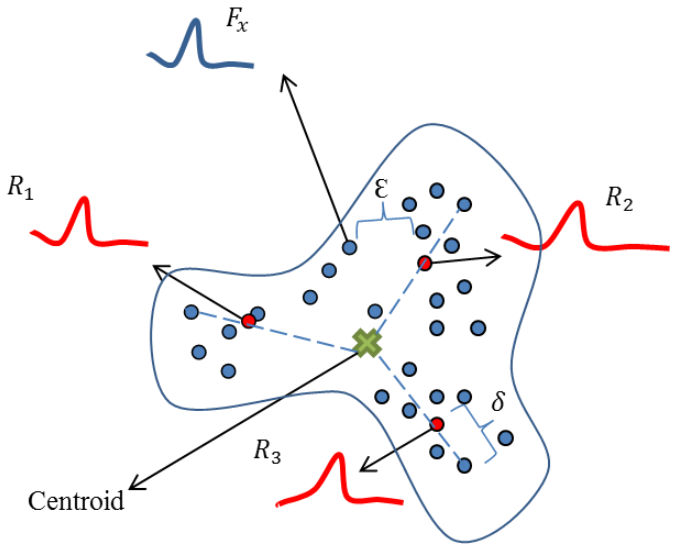


Figure 4.17: Multi-representative approach for time-series clustering

However, the multi-representative method used in this study for time-series clustering differs from CURE approach in two aspects:

1) Using the centroid as one of the representatives (beside the well-scattered time-series data) which help to find better results.

2) Using a dynamic number of representatives rather than CURE algorithm which uses a constant number of representatives for each cluster.

In a cluster of time-series data, it is obvious that representing a cluster with lowest number of representatives is ideal. To find the representatives dynamically, given a set of time-series, a threshold is defined for representatives as minimum density, Ɛ, such that the distance of each time-series in the cluster from its corresponding representative, is less than Ɛ. That is, adding the representatives to sub-cluster, is continued until it is guaranteed that all time-series are close enough (Ɛ) to at least one representative of sub-cluster. It leads to find the representatives based on their shape, not based on a fix number of representatives. The representatives of the sub-clusters are calculated using the following algorithm (Figure 4.18):

Input: SC: All time-series in sub-cluster

Ɛ: Min density

α: Shrink factor

Output: R: A set of representatives

Method: Find_representatives (SC, Ɛ, δ)

1. R=centroid(SC); /* using the equation 4.17*/
2. If δ >0 /* using multi-representative approach */
3. max_dis= Max{distance(F_x, R): $F_x \in$ SC}
4. while max_dis> Ɛ

   /* Add Representative: let F_x be the farthest object from set R */

5. $F_r = F_x : Max_x \{dis_{ED}(F_x, F_y), F_y \in R, F_x \in SC\}$
6. SC = SC $- F_x$;
7. F_r=((1- δ)* F_r+ δ * centroid(SC)) /* to shrink toward centroid */

8. R = R ∪ F_r;

9. max_dis= Max{dis_{ED} (F_x, R): F_x ∈ SC}

10. end while

11. end if

12. return R

Figure 4.18: Algorithm of determining representatives in sub-clusters

4.5 Step 3: Merging

The output of second step is relatively few number of prototypes (in comparison to original dataset), as representatives of sub-clusters. Some sub-clusters were represented by only one representative, some by more than one. Now, the prototypes are combined to form the final clusters. Considering each prototype as one time-series, the process of merging is performed to group the prototypes. Different approaches (partitioning, hierarchical, etc.) can be adopted for merging the sub-clusters (or clustering of prototypes). Essentially, choice of clustering algorithm depends both on the type of desired clusters (arbitrary or spherical), and on the particular purpose and application of clustering (Warrenliao, 2005). If the size of clusters are equal (comparable size), clusters are hyper-ellipsoidal (or globular or spherical). In this case, k-Medoids or hierarchical algorithm with complete or average linkage are the best choice to cluster time-series, because they try to minimize the distortion in data (Chaoji, Al Hasan, Salem, & Zaki, 2008). In contrast, in some datasets (such as image segmentation or spatial data mining) where clusters are of different densities (Chaoji et al., 2008), algorithms for making arbitrary shape clusters are used. Some conventional algorithms which try to find arbitrary clusters in datasets are DBSCAN, CURE, ROCK and CHAMELEON. The readers are referred to (Karypis et al., 1999) for comparison and references.

Moreover, if the clusters are very close to each other or are from different density and size, different type of algorithms (used for merging) will result in different cluster structures. Therefore, choosing the proper algorithm is very critical. MTC works with prototype/prototypes (where are very small in compare to whole data), as a result, different type of clustering algorithms can be adopted, consequently, different type of clusters can be made depending on problem in hand. Figure 4.19 shows the overview of this process.

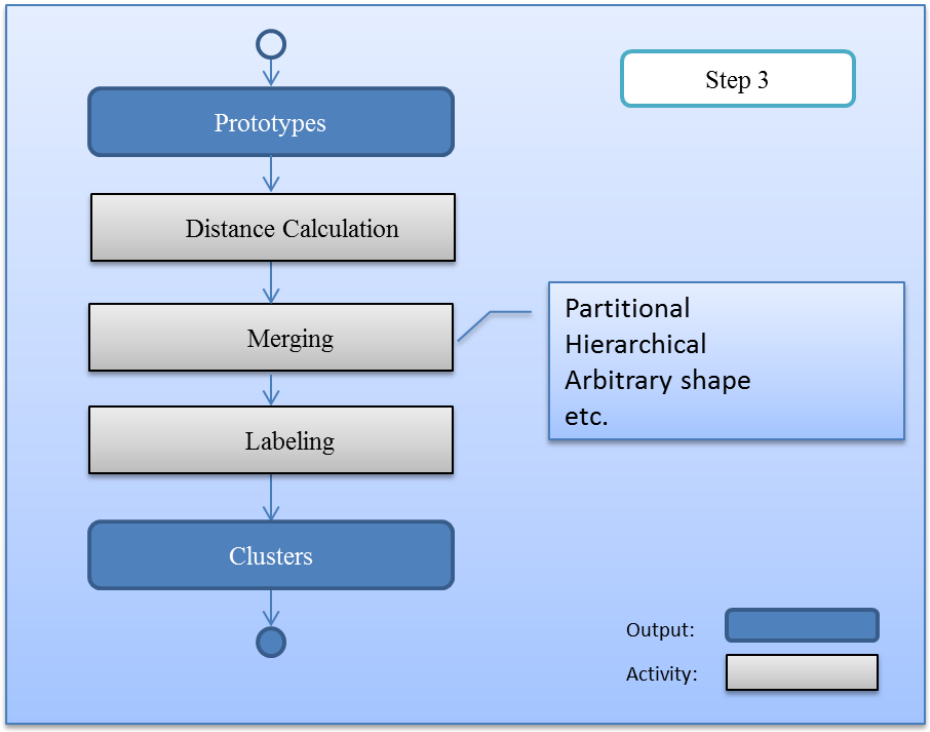


Figure 4.19: The workflow of MTC clustering for third step.

4.5.1 Activity1: Distance Calculation

In the first two steps of MTC, similar time-series in time were grouped precisely. However, two time-series which are similar in shape may not be similar in time. Hence, in this activity the similarity in shape of time-series are computed. As explained in 2.5.2, Euclidean distance requires that the two time-series must have one-to-one alignments, which sometimes is not desirable if one is looking for shape based similarity between time-series that have discrepancy in x-axis values as it is depicted in Figure 4.20.

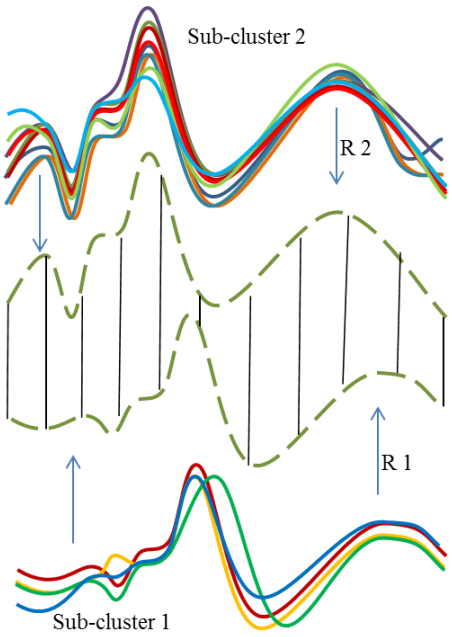


Figure 4.20: Similarity in time between sub-cluster's prototypes

In contrast to similarity in time, DTW (Dynamic Time Wrapping) will calculate all possible mappings between two time-series' data points, finding the smallest possible warping distance. As a result, in the third step of MTC, DTW is desirable because it can distinguish the time-series which are different from other time-series in the cluster in terms of similarity in shape. The accuracy of DTW is calculated in front of ED in Section 5.3.3.1. Although DTW is computationally expensive, it is not adopted on all dataset, and it is used to measure only the similarity on a subset of whole dataset (prototypes), so, it can be run very fast. Moreover, it can be applied by some constraints which decrease its complexity as it is explained in Section 4.7. Figure 4.21 depicts the intuition of using DTW in the third step of MTC. As this figure shows, the similarity between the time-series inside the sub-clusters is computed based on the similarity in time (within similarity). However, the similarity between the prototypes is calculated based on similarity in shape.

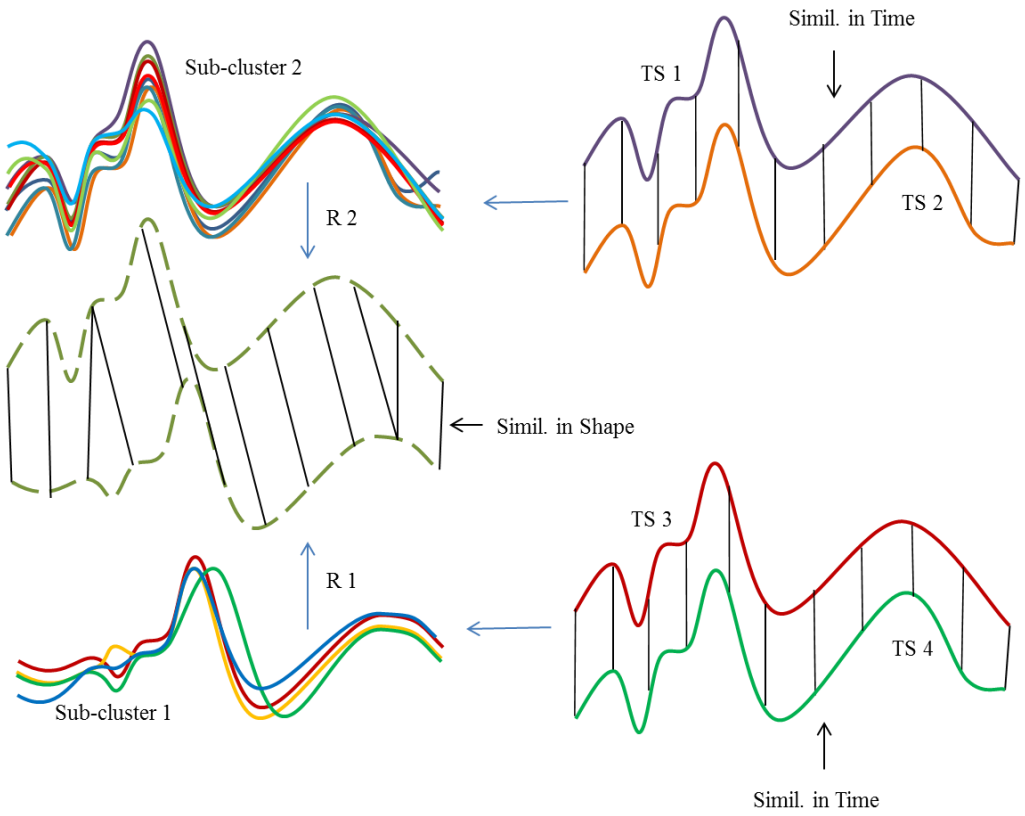


Figure 4.21: Intuition for using DTW for calculating similarity in shape between representatives of sub-clusters in the third step of MTC

Therefore, to calculate the distance between two sub-clusters, the DTW of their prototypes are calculated. Suppose $R_x = \{r_{x1}, \dots, r_{xi}, .., r_{xT}\}$ prototype of SC_x where r_x is calculated by Equation 4.17. Then, to compute the distance between prototype of SC_x and SC_y , a n × n matrix is constructed for distance of all matching pairs as M(R_x, R_y) where $M_{i,j} = dis_{ED}(r_{xi}, r_{yj})$ and $dis_{ED}()$ is calculated by Equation 2.2. Given $W = \{w_1, w_2, \dots, w_u\}$ as a set of warping paths where w_u = {(r_{x1}, r_{y1}), (r_{xi}, r_{yj}), . . . , (r_{xT} , r_{yT})} is a set of points that define a traversal of matrix M, DTW between two prototypes R_x and R_y is a warping path that minimize the distance between R_x and R_y.

$$dis_{DTW}(R_x, R_y) = \min(\sum_{u=1}^{U} W_u/U) \qquad 4.18$$

where (r_{x1}, r_{y1}) = (1, 1) and (r_{xT}, r_{yT}) = (n, n), and that $0 \le r_{xT+1} - r_{xT} \le 1$ and $0 \le r_{yT} - r_{yT+1} \le 1$ for all $T < n$.

4.5.2 Activity2: Merging Sub-clusters

As mentioned, MTC covers different schemes of clustering. In the following sub-sections three different approaches of merging sub-clusters are explained which leads constructing various structures of clusters.

4.5.2.1 Merging by Partitioning Clustering

For merging the sub-clusters, the mechanism of each partitioning clustering algorithm can be used, such as k-Medoids, k-Means, etc. The k-Means algorithm uses centroids as cluster representatives, and the k-Medoids algorithm takes its representatives from the original data. However, unfortunately, k-Means algorithm which needs averaging for prototype, will sometimes fail to give correct results especially in the case when Dynamic Time Warping (DTW) is used as the distance measure in averaging the shape of time-series (Niennattrakul & Ratanamahatana, 2007). Therefore, the approach used in this thesis to define the representative for partitioning clustering is the medoid of each cluster. Accordingly, the medoid of a cluster, C_{j} , is defined as:

$$Medoid_{C_{\mathrm{j}}} = R_x : Min_{F_x} \{ \sum_{R_{\mathrm{y}} \in C_{\mathrm{j}}} dis_{DTW}(R_x, R_{\mathrm{y}}), R_x \in C_{\mathrm{i}} \} \qquad 4.19$$

Thus a "medoid" is a time-series that best represents a set of prototypes while those prototypes are themselves representing more time-series (sub-clusters). Although, it is not a linear-time algorithm for clustering (the initial construction of the pairwise distance matrix *Dis* requires time $O(dn^2)$ and the search for a new medoid (each iteration) takes time $O(n^2)$), because the number of objects, n, are the number of representatives which is quite small, it is practical for large datasets. Pseudocode for k-Medoids is given in Figure 4.22. Inputs of this algorithm are *Dis*, the $n \times n$ matrix of DTW distances between all pairs of time-series, the number of medoids desired *k*, and the initial medoids.

Input: $Dis \in R^{n \times n}$ the distance matrix

$R \in R^n$ /* the set of representatives improrted from second step */

K: number of desired clusters

Medoid: Initial medoids

Output: C /* clusters of representatives */

Method: k-Medoids (Dis, K, Medoid, R)

1. while any $Medoid_{C_j}$changes do
2. for i ∈ {1 . . . n} do /* n is the number of TS (prototypes of sub-clusters) */
3. C(i) ← arg $min_j Dis_{ij}$ /* find the closest medoid F_j to F_i*/
4. end for
5. for j ∈ {1 . . . k} do
6. $Medoid_{\mathrm{j}} \leftarrow R_x : Min_{F_x} \{\sum_{R_{\mathrm{y}} \in C_{\mathrm{j}}} dis_{DTW}(R_x, R_{\mathrm{y}}), R_x \in C_{\mathrm{j}}\}$
7. end for
8. end while
9. return C

Figure 4.22: Pseudocode for the k-Medoids algorithm.

Partitioning clustering of time-series data is very good as summarization or as a pre-processing phase for more complex data mining tasks. However, considering the clustering as a data understanding tool, finding the hierarchical clusters is more meaningful and intuitive (because has the ability to show the results as a tree of clusters). In the following (Section 4.5.2.2), merging of sub-clusters using a hierarchical clustering is explained.

4.5.2.2 Merging by hierarchically clustering

As discussed in Section 2.8.1, the main problem of using hierarchical methods for time-series clustering is its inability to scale well—the time complexity of hierarchical algorithms is at least $O(dN^2)$ (where N is the total number of instances and d is the number of dimension of TS), which is non-linear with the number of time-series.

Moreover, clustering of a large number of time-series using a hierarchical algorithm is also characterized by huge I/O costs. However, in MTC, this problem does not exist anymore because the number of time-series in the third step is small (it is equal to number of representatives), and therefore, the advantages of hierarchical clustering come to account including:

- Versatility: covering different linkage methods enable MTC with hierarchical scheme to make different shaped of clusters. The single-link methods, for example, maintain good performance on datasets containing non-isotropic clusters, including well separated, chain-like and concentric clusters (see Section 5.3.3.2).
- Multiple partitions: hierarchical methods produce not one partition, but multiple nested partitions, which allow different users to choose different partitions, according to the desired similarity level. The hierarchical partition is presented using the dendrogram. Dendrogram creates a decomposition of the given data and provides an intuitive view for users which are very useful especially in large datasets. For example, MTC can produce an intuitive dendrogram of companies (based on their stock) which is very desirable as depicted in Figure 6.25. It is important to notice that in MTC, unlike other methods, all time-series are not contributed for constructing the dendrogram, instead, only prototypes are used.

Considering the advantages of hierarchical clustering, each kind of hierarchical clustering can be used for clustering of prototypes (e.g., single linkage, complete linkage, etc.). These approaches are very simple and given representatives can be easily applied to make the clusters. However, in the following, to show the scale of the MTC, a hierarchical based algorithm is proposed for clustering of representative, which works on non-spherical data. This scheme works with sub-clusters represented by multi-representative to generate the tree of nested and arbitrary shape clusters.

4.5.2.3 Merging for arbitrary shape clusters

A good clustering algorithm should identify arbitrary shaped clusters (Gordon, 1999). In arbitrary shape clustering, clusters should be separated by distances larger than the intra-distance of clusters. Even good separated centroids in partitioning clustering, do not necessarily imply good separated clusters. Now the question is whether time-series clusters are arbitrary or spherical? The problem of centroid-based clustering also happens for time-series data where a time-series in a cluster is closer to centroid of another cluster than to its natural cluster. It occurs especially when there are clusters with different sizes, sparse time-series, or outlier time-series. Since the sub-cluster sizes are usually different, their representatives also are sparse. This is exactly such as what happens in 2-dimensional objects seen in Figure 4.23. As the figure shows, in some datasets, the sub-clusters (their representatives) of a natural cluster may be very far apart each other (or not be spherical around the centroid) which leads to split the cluster. Consequently, a partitioning clustering may perform poorly in these situations.

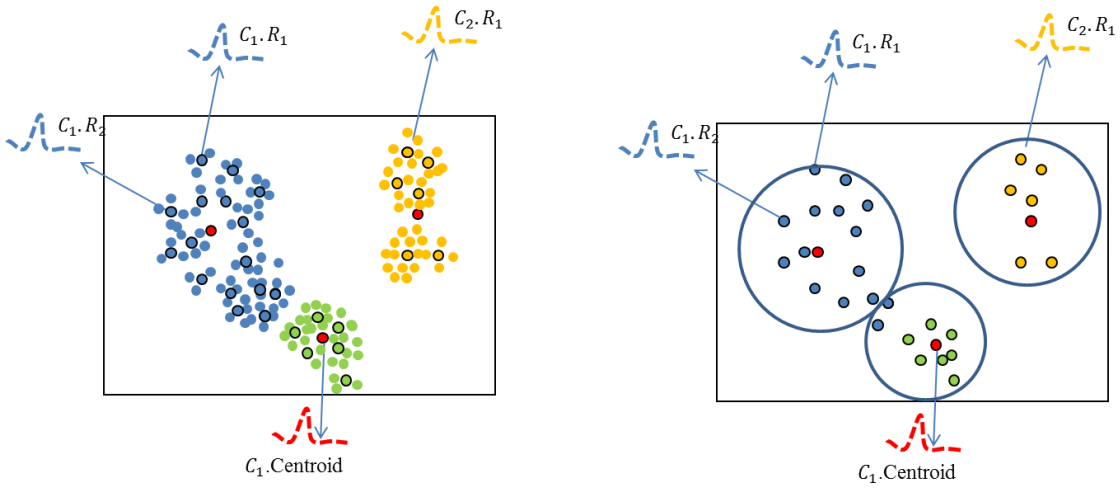


Figure 4.23: Splitting the natural cluster due to incorrect spherical clustering

It is shown in experimental results (see Section 5.3.3) that some time-series dataset have arbitrary shape, due to their diverse nature (which is of the spatial object's characteristics). Other than that, time-series datasets include outlier time-series. Outliers are time-series data which are not belong to any cluster and should be discarded during

clustering process. A clustering algorithm should have the capability of the handling of noise and outliers. Generally, arbitrary based clustering algorithms handle outliers.

Arbitrary shape clustering algorithms were proposed in several works. The most successful among others are Spectral (Shi, 2000), density-based, e.g., DBSCAN (Ester et al., 1996), and nearest neighbour graph based, e.g., Chameleon (Karypis et al., 1999) approaches. The problem of these algorithms is their poor scalability (Chaoji et al., 2008). As a result, the existing approaches cannot be applied directly on time-series datasets because they are very large and high dimensional.

In this activity, it is shown that MTC can make arbitrary shape clusters using multi-representative approach. In this approach, the fine-clustered time-series (sub-clusters) are merged using an agglomerative hierarchical clustering. As mentioned, each sub-cluster can be represented by multi-representatives. Now, the most similar representatives are merged. Each set of representatives in the sub-cluster is treated as members of a formed cluster through the merging process. Then, the merging activity is performed in such a way that prototypes of a sub-cluster should remind in the same sub-cluster in final clusters.

There are essentially different approaches to find the most similar clusters and merging them. It can be object closeness (it is same as single, complete, or average linkage in hierarchical clustering), centroid closeness such as WPGMA, WPGMC (Sneath & Sokal, 1973), relative inter-connectivity, or relative closeness such as CHAMELEON (Karypis et al., 1999). In this algorithm, the object closeness is used to find the most similar clusters. Figure 4.24 depicts the pseudocode for merging sub-clusters of time-series using the representative closeness of each sub-cluster.

Input: $SC = \{sc_1, sc_2, .., sc_i\}$ set of sub-clusters

$sc_i.R_{ik}$: $the\ k_{th}\ representative\ of\ sc_i$

K: number of desired clusters

Output: SC /* merged sub-clusters */

Method: Cluster (SC,k)

1. for all R_{ik}, R_{jz}
2. If i=j /* representatives belong to the same sub-cluster */
3. $A_{ik,jz}$= 0; /* a distance matrix for representatives */
4. else
5. $A_{ik,jz}$=$dis_{DTW}(R_{ik}, R_{jz})$/* calculate the distance between all representatives, and store the results in a distance matrix. */
6. end IF
7. end for
8. while (size(SC)>k)
9. [i, j] = $\min\left(A_{ik\times jz}\right), \forall i, j | \ i \neq j$; /* Find closest sub-clusters by finding closest representatives of mutual exclusive sub-clusters */
10. $sc_i = sc_i \cup sc_j$ /* Merge cluster j into cluster i */
11. $sc_i.R = sc_i.\mathrm{R} \cup sc_j.\mathrm{R}$
12. $A_{ik,jz} = 0$ /* Update distance matrix by removing the distance among representatives of merged clusters */
13. $\mathrm{SC} = \mathrm{SC} - sc_j$ /* Delete cs_j */
14. end while
15. return SC

Figure 4.24: An outline of clustering algorithm for clustering of sub-clusters with multi-representatives

As mentioned, the output of the second step of the algorithm is a set of sub-clusters, $SC = \{sc_1, sc_2, .., sc_i\}$, where the number of sub-clusters is a relatively small (in comparison with the original data) and each one is represented by a set of prototypes,

$sc_i.R$. During the third step of the algorithm, at first a similarity measure for each pair of seed clusters, R_{ij}, is computed. Then, the distance between the representatives which are from the same clusters are initialized to zero (see lines 1-6 in Figure 4.24). It leads to not separating the representatives of the same class to different clusters during merge process. The similarity between clusters is then used to merge the two clusters having the minimum distance. The similarity among clusters can be defined based on any clustering algorithm that can use the similarity function to merge the clusters. For example, single link or complete link method can be applied on data to define the distance between two clusters for sake of simplicity. The single (complete) linkage algorithm measures the similarity between two clusters as the similarity of the closest (farthest) pair of time-series belonging to different clusters. Merging the clusters means combining their representatives. The algorithm repeats the merging process until all the objects are eventually merged to form one cluster (or until the desired number of clusters is obtained or the distance between two closest clusters is above a certain threshold distance). As a result, a hierarchy of sub-clusters is generated.

This algorithm finds the arbitrary shape clusters and is robust to outlier time-series. As mentioned, outliers refer to time-series which are not contained in any cluster and should be discarded during the mining process. After constructing the pre-clusters in the first step, and decomposing them in the second step, some clusters with a few time-series are observed (beside rather large clusters). These clusters are clusters which are potential outliers or a part of a cluster which were grouped in the first step incorrectly, and then detected, and separated as a new cluster in the second step. Considering the outlier characteristic, outliers do not tend to merge with other clusters because of their larger distance in comparison with other time-series. Hence, the small clusters which are not merged till last iterations of merging can be discarded.

4.5.3 Activity3: Mapping

The result of merging process is some clusters which are constructed from prototypes of sub-clusters. Thus, a mapping activity is carried out to assign the original time-series to their correspond prototypes. All time-series of each sub-cluster are assigned to the cluster which the corresponding prototype is assigned.

4.6 MTC Algorithm

In the following, the outline of the all steps of MTC is shown in Figure 4.25.

Input:

D: A time-series dataset /* $D = \{F_1, F_2, .., F_n\}$ */

WL: SAX window level

WS: SAX word size

K:cluster count

Output: C

Method: MTC (*D, WL, WS, K*)

/* Step-1 time-series pre-clustering */

1. *D* =z-norm(*D)* /* z-normalization of all time-series */
2. for each time-series *i*
3. $\hat{F}_i$ = SAX_Transform(F_i, WL ,WS)
4. $\hat{D} = \hat{D} \cup \hat{F}_i$
5. end for
6. *Apx_sim*[*n*][*n*]= $dis_{APXDIST}$ ($\hat{D}$) /* Find similarity array */
7. pre-cluster[1 to *n*]=Ek-Modes(*Apx_sim*[*n*][*n*],k) */ see Figure 4.10 */

/* Step-2 cluster purifying */

8. for *i* = 1 to *k*; *k* is the group count of pre-clusters
9. *ts_cnt* = the time-series count in pre-cluster *i*
10. *TS_TMP*= *D*(pre-cluster [1 to *n*]==*i*)
11. *sim[i]* [*ts_cnt*][*ts_cnt*] =1-dis_{ED} (*TS_TMP*) /* similarity matrix of all TS in pre-cluster *i*

```
    by ED */
12.     sub-cluster[1 to ts_cnt]=PCS(sim [1 to ts_cnt])  /* see Figure 4.14  */
13.     rep[i] = find_representatives(sub-cluster) /* see Figure 4.18 */
14. end for
/* Step3 merging */
15. C_rep= Cluster (rep[i],k)   /* clustering of representatives, using an arbitrary scheme */
16. C=Replace_mem (C_rep, D)   /* mapping process */
17. validate the MTC clustering results
18. return C
```

Figure 4.25: Pseudocode of MTC model

4.7 Interactive MTC (IMTC)

MTC runs on datasets as a batch clustering algorithm focusing on the accuracy of final clusters. However, in the case that a user needs to see a primitive or meantime results (or cannot stand till finishing of batch), MTC is run as an interactive model. That is, the algorithm starts running, generates a primitive result (not accurate), and keep improving it gradually until user interrupts the process. In moment of interruption, the best result so far, is generated and is shown to the user. Figure 4.26 shows the steps of IMTC. As the diagram shows, the output of third step is passed again to second step for more revising.

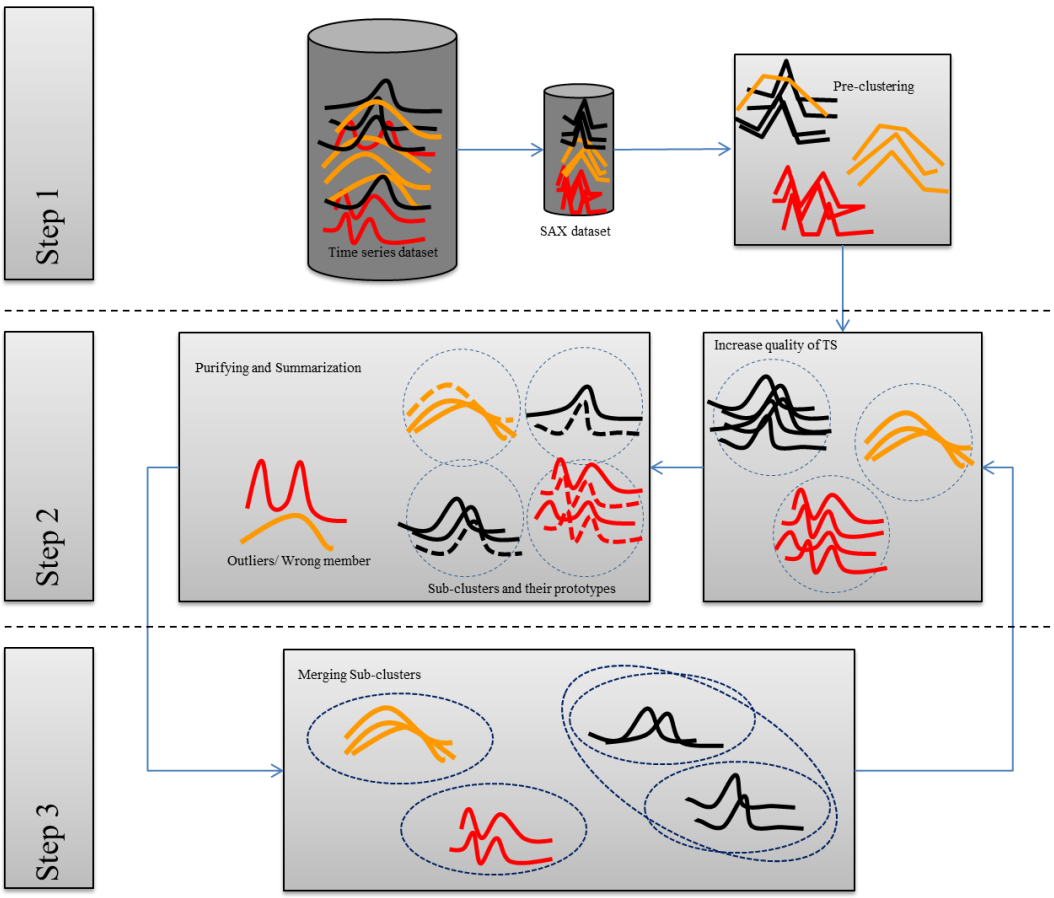


Figure 4.26: An overview of IMTC

Exploiting multi-step characteristic of MTC, IMCT can be carried out as an interactive clustering algorithm. As mentioned, in this approach, an inaccurate result is constructed using a low resolution time-series (in step-2) and a very narrow constraint of DTW (in step-2). Then, step-2 and step-3 are performed alternatively. However, in step-2, only one cluster is candidate to be refined (dispersed) in each iteration. Selecting the candidate cluster is based on within cluster variance (the criteria defined by the mean and standard deviation of affinity of clusters). This process is continued until all pre-clusters are purified at least one time. In the process of purifying the candidate cluster, the purifying is performed more precisely in comparison with previous iteration. That is, in each iteration, the measurement improves (for the candidate cluster) because firstly the resolution of time-series that exist in the candidate cluster increased by using a lower compression-ratio of time-series. Second, the constraint of DTW used in the third level, is widened through each iteration. The whole process is same as split and

merge and meanwhile increasing the accuracy of calculations. This process continues until first, all low resolution time-series are replaced with a high resolution time-series (or with original time-series). Second, the distance between all pairs are calculated by complete DTW (without constraint), and finally, all pre-clusters have been checked for purifying process at least one time. This process results in improving the quality of clustering over time. Figure 4.27 shows the design of overall process.

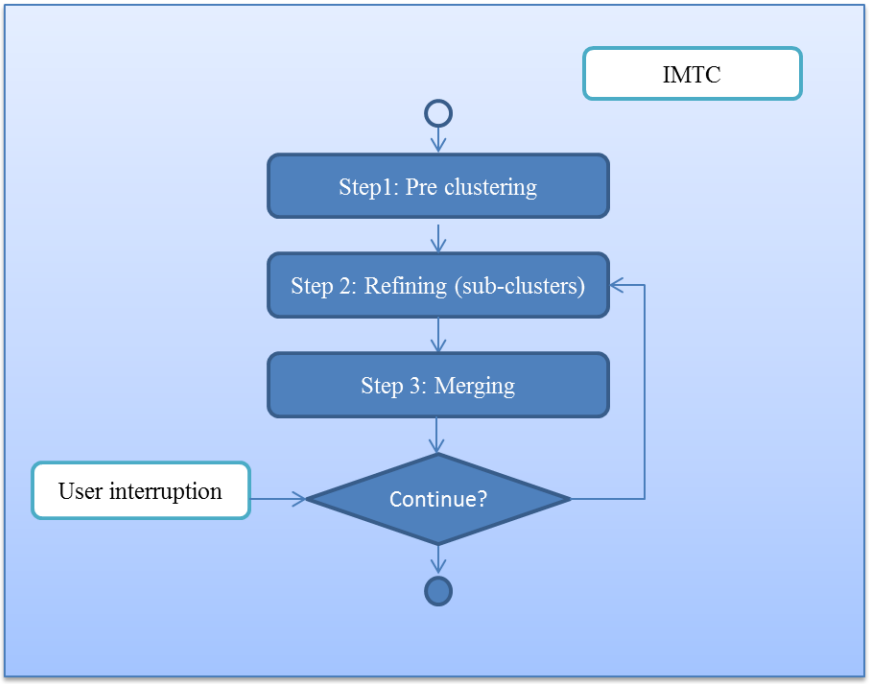


Figure 4.27: The activities of IMTC model

As mentioned, DTW distance measure is used in the third step of IMTC same as MTC model. However, in IMTC, DTW is applied by some global constraints (Itakura, 1975; Sakoe & Chiba, 1978) which decreases its complexity. As mentioned in 2.5.3, DTW is the warping path that minimizes the total distance between two series of F_i and F_j. Global constraints determine the amount of warping allowed to find the minimum path. The width of warping window is shown by parameter DTW_r. Euclidean distance can be seen as an upper bound of DTW. That is, Euclidean distance is a special case of DTW with a very narrow constraint, where the warping path in calculation of DTW is exactly the diagonal of matrix. However, the constraint can be widened which means how far

warping path may stray from the diagonal. The widest constraint (i.e., without constraint) means the complete DTW, i.e., $DTW_r = d$ where d is the length of time-series. Constraints provide a property for DTW where accuracy of similarity in shape between two time-series can be calculated approximately with low complexity. This property of DTW, i.e., applying global constraints (Itakura, 1975; Sakoe & Chiba, 1978), is used in IMTC to calculate the distance between time-series cheaply and then improve it. It means whichever MTC is running as an interactive clustering, the DTW with a very narrow constraint (Euclidean distance) is used where $DTW_r = 0$, and then accuracy of metric can be gradually increased by widening the constraint to complete DTW, i.e., $DTW_r = d$ or $DTW_r = 100\%$.

4.8 Chapter Summary

In this chapter, the multi-step clustering model proposed in previous chapter were explained and designed in details. In the pre-clustering step, dimension of time-series data was reduced by SAX technique. SAX representation was used to cluster the low-resolution time-series which can fit in memory rather than original (raw) time-series dataset. Then, a new distance measure compatible with SAX representation was developed to construct pre-clusters (APXDIST). This approach is accurate for clustering, because based on its definition, it considered the distance of neighbour symbols as a non-zero value, and does not ignores the maxima and minima points of time-series. Subsequently, a new clustering algorithm (Ek-Modes) was proposed for clustering of approximated time-series. The output of this step was pre-clusters which were passed to second step. Then pre-clusters were purified in the second step by PCS (Purify Cluster Search). As mentioned, an affinity threshold was defined to create sub-clusters of high affinity time-series. It was defined dynamically which is very important. The output of second step was some sub-clusters which were represented by their prototypes. That is, , instead of applying the clustering algorithm to the entire data, only

the prototype/prototypes of each sub-cluster are participated in merging process which reduces the complexity of process to a high extend. Subsequently, in the third step, depend on the problem in hand, three clustering scheme was proposed to merge sub-clusters. Finally, the IMTC which is the interactive model of multi-step was explained in detail. It was explained that IMTC is advantageous once a user needs to see a primitive or meantime results.

5.0 EXPERIMENTAL RESULTS AND ANALYSIS

5.1 Introduction

In this chapter, datasets used for evaluation of proposed models are introduced. Then, all the methods and algorithms designed to achieve the objectives of each step are experimentally discussed in this chapter. The performance of APXDIST and Ek-Modes are shown on different datasets. Then, PCS is applied on a range of datasets to show the obtained purity in the second step of MTC. Consequently, the observations are analysed to show how the results are generated using MTC and how these methods increase the accuracy of final clusters. Details of each dataset and generating procedure of each synthetic data are explained in Section 5.2. Each step of MTC model is analysed in Section 5.3. Then the final results gained by applying MTC on the dataset are reported in Section 5.4. Later, the final results are discussed extensively in the next chapter.

5.2 Datasets

The proposed model is experimented with 21 different datasets in various domains and sizes. 19 of these datasets are from the UCR Time-series Data Mining Archive (Keogh & Folias, 2002). The size of time-series in these datasets ranges from 28 to 6,000 records, and their exact size is indicated in Keogh & Folias (2002). This set is chosen because it is of various numbers of clusters, different cluster shapes and density, contains noise points, and used in many articles in the literature as a benchmark. These datasets have two sets in the repository, namely TRAIN and TEST. In this study the TEST set is used because it includes large datasets and the TRAIN sets is used to visualize the results for the sake of simplicity. In the set of datasets used in this thesis, some of datasets which contain 1,000 to 12,000 time series (each time series may have the length 100 to 700 points) are considered to experiment MTC on large time series data sets. These data sets are considered large because for example one of the largest

dataset in the literature which includes 9,236 time-series take about 127 days using a batch algorithm which is a severe bottleneck in clustering of time series (Q. Zhu, Rakthanmanon, Batista, & Keogh, 2012). However, some small sets also are applied for visualization purpose or to show the behaviour of proposed model (MTC) on small datasets. All these 19 datasets have class labels and can be used for evaluation of MTC using external index. Moreover, another two datasets are chosen from real-world problems, namely Stock Exchange of Malaysia (KLSE) and Account Balance of customers of a Malaysian Bank (BTD), to show the application of the proposed model in different domains. Class labels of these datasets are unavailable and internal index is used to evaluate the accuracy of the clusters.

Table 5.1: Number of clusters, number of instances and the length of the time-series in each dataset

	Dataset	**Clusters**	**Instances**	**Length**	**Class label**	**Type**
1	50words	50	455	270	Yes	Real-world
2	Adiac	37	391	176	Yes	Real-world
3	CBF	3	900 (up to 6000)	128	Yes	Synthetic
4	Coffee	2	28	286	Yes	Real-world
5	ECG200	2	100	96	Yes	Real-world
6	FaceAll	14	1690	131	Yes	Real-world
7	FaceFour	4	88	350	Yes	Real-world
8	FISH	7	175	463	Yes	Real-world
9	Gun_Point	2	150	150	Yes	Real-world
10	Lighting2	2	61	637	Yes	Real-world
11	Lighting7	7	73	319	Yes	Real-world
12	OliveOil	4	30	570	Yes	Real-world
13	OSULeaf	6	242	427	Yes	Real-world
14	SwedishLeaf	15	625	128	Yes	Real-world
15	synthetic_control	6	300 (up to 12000)	60	Yes	Synthetic
16	Trace	4	100	275	Yes	Real-world
17	Two_Patterns	4	4000	128	Yes	Real-world
18	Wafer	2	6164	152	Yes	Real-world
19	yoga	2	3000	426	Yes	Real-world
20	KLSE	-	870	242	No	Real-world
21	BTD	-	6802	360	No	Real-world

Some of these datasets are related to real-world problems, and some include syntactic datasets. For the syntactic dataset, up to 10,000 records are generated to experiment on large datasets. Table 5.1 shows the utilized datasets for the experiments, so the results can be compared across different algorithms.

5.2.1 Real-world Dataset

The real world datasets selected from the UCR repository have been tested for clustering by other researchers and used as benchmarking for comparison (Keogh & Kasetty, 2003). To show the associative property, some of these datasets are explained in following subsections. Leaf, Face, Gun are of type of multimedia data transformed into time-series (Ratanamahatana, 2005). ECG is from health care domain and related to electrocardiograms of patients.

Leaf Dataset: The Leaf dataset contains the leaf images of 6 different leaf species (classes) where raw image shapes are transformed to time-series data by image processing technique as illustrated in Figure 5.1. This dataset holds 242 time-series of lengths 427 data points. Class 1 through class 6 contains 34, 29, 33, 53, 36, and 38 instances, respectively. Figure 5.2 shows some examples of the data from each class.

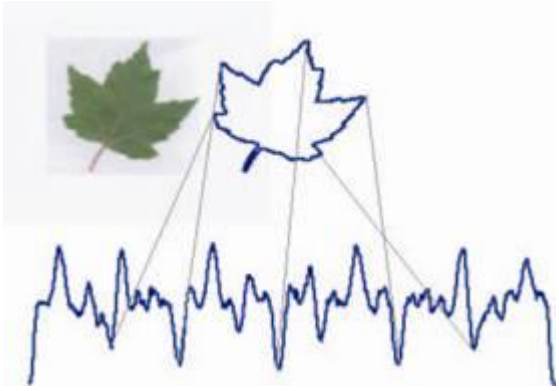


Figure 5.1: Transformation of raw image shape of a leaf to time-series data (Ratanamahatana & Niennattrakul, 2006)

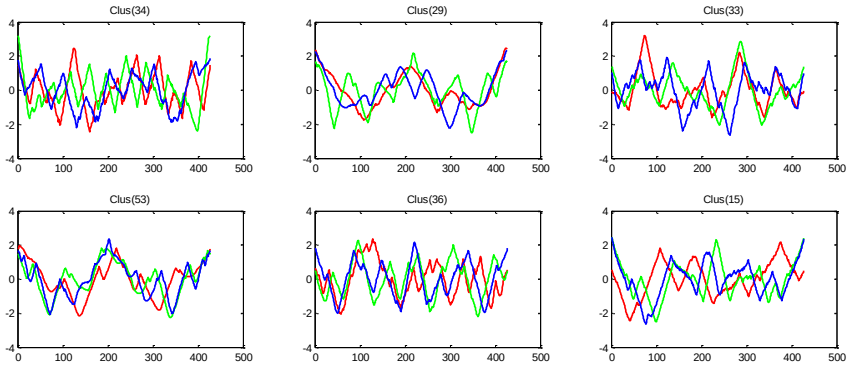


Figure 5.2: Three sample time-series of six classes (six species) of Leaf dataset

Face Dataset: Face dataset includes different face expressions of four head profiles (related to four persons). This dataset contains 88 time-series of head profiles. The length of each time-series is between 107 to 240 data points as illustrated in Figure 5.3. All time-series in this dataset are normalized to the same length of 350 data points. Three examples of each class of face profile (which is related to the four individuals) are depicted in Figure 5.4. This dataset is used also for multimedia clustering (Ratanamahatana & Niennattrakul, 2006).

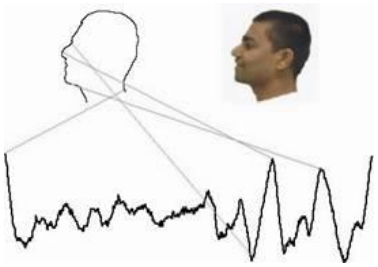


Figure 5.3: Transformation of the image of a head profile to time-series data (Ratanamahatana & Niennattrakul, 2006)

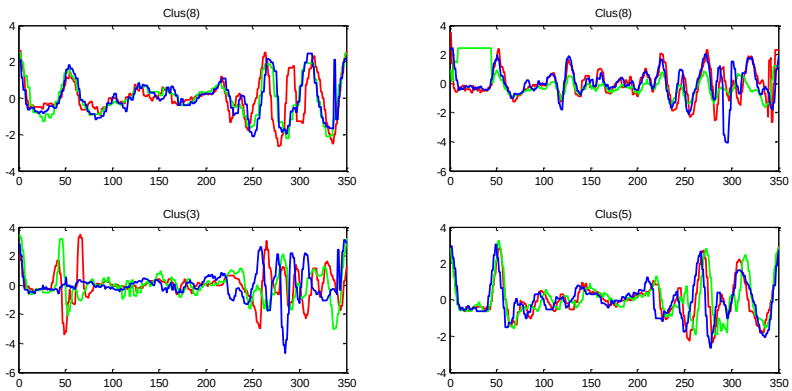


Figure 5.4: Examples of four different faces

ECG Dataset: The ECG dataset includes time-series of two different heart pulse classes, i.e., normal and abnormal. Each class includes 100 instances of length 96 data points, and have been normalized in this experiment. This dataset is used also in (Ratanamahatana & Niennattrakul, 2006). Figure 5.5 shows some examples of the data.

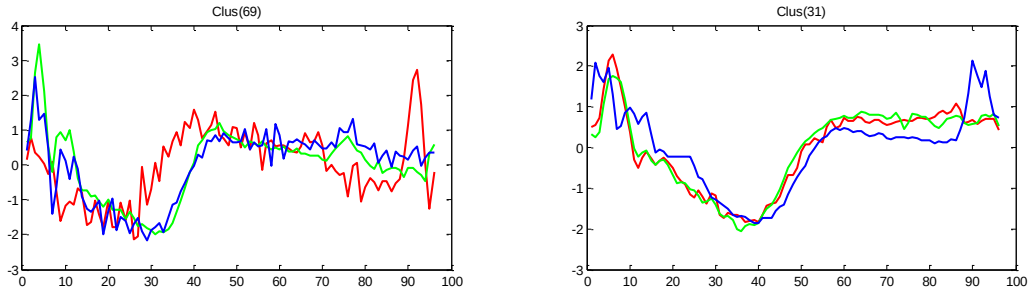


Figure 5.5: Three examples of two classes of ECG dataset (left: 'normal', right: 'abnormal'.)

Gun Dataset: Gun dataset consists of two classes of time-series obtained from the video surveillance domain, by plotting the movement of a hand with a gun and without the gun in every video frame using some image processing techniques (Ratanamahatana & Keogh, 2004b). Each class includes 150 instances, and the length of each instance is 150 data points. Figure 5.6 shows some examples of the two-class Gun dataset.

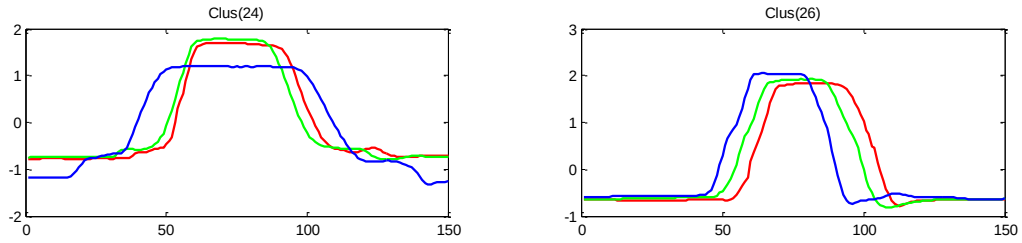


Figure 5.6: Three sample time-series of two classes of gun dataset

5.2.2 Synthetic Datasets

In the following subsections two synthetic datasets used in the experiment are introduced. These datasets are used to show the scalability of system on large datasets.

Cylinder–Bell–Funnel (CBF): The well-known 3-class CBF dataset contains 900 instances with the length of 128 data points. The Cylinder-Bell-Funnel (CBF) dataset is a syntactic dataset (artificial) which has the properties of temporal domain and was

originally proposed by Saito (2000) and then used by many works (J. Lin et al., 2007; H. Zhang et al., 2006; X. Zhang et al., 2011). This dataset is one of the most commonly used dataset for time-series classification and clustering experiments. This dataset includes three types of time-series: cylinder (c), bell (b) and funnel (f). In order to show the scalability of MTC, different cardinalities of CBF are generated and used in this study. The instances are generated using random functions as follows (H. Zhang et al., 2006).

$$\begin{aligned} c(t) &= (6+\mu).X_{[a,b]}(t) + \varepsilon(t) \\ b(t) &= (6+\mu).X_{[a,b]}(t-a)/(b-a) + \varepsilon(t) \\ f(t) &= (6+\mu).X_{[a,b]}(b-t)/(b-a) + \varepsilon(t) \end{aligned} \quad 5.1$$

and

$$X_{[a,b]} = \begin{cases} 0, & if\ t < a \vee t > b \\ 1, & if\ a \leq t \leq b \end{cases} \quad 5.2$$

where *t=1,..,128* and $X_{[a,b]}$ is the characteristic function. μ is drawn from a standard normal distribution *N(0,1)* and cause random amplitude variation in time-series. Some random noise inserted in time-series by $\varepsilon(t)$ which is drawn from a standard normal distribution *N(0,1)*. The value of a is drawn uniformly from the range [16; 32] and (*b-a*) is an integer-value drawn uniformly distribution from the range [32; 96]. Since b-a can vary from 32 to 96, there is significant temporal variation in the duration of events. Moreover, the start of events also varies because of variation of *a* from 16 to 32. Examples of CBF time-series are shown in Figure 5.7.

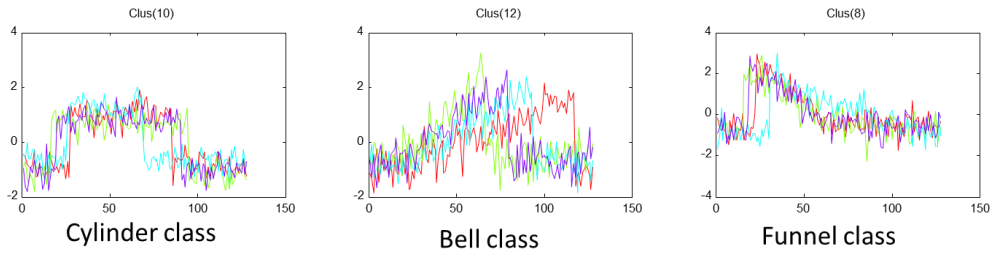


Figure 5.7: Four samples of each class of Cylinder, Bell, and Funnel (CBF) dataset

Control Chart (CC): Control Chart Time-series (CC) is a synthetic dataset proposed by Alcock & Manolopoulos (1999) and was used in many researches including classification (Geurts, 2001) and clustering (J. Lin, Keogh, Lonardi, et al., 2003; J. Lin et al., 2007; X. Zhang et al., 2011). This dataset has six classes and 300 time-series. Figure 5.8 depicts three sample time-series of each class where each series is produced by:

$$f(x) = \begin{cases} m + rs, & if\ c(o) = Normal \\ m + rs + \text{asin}(\frac{2\pi t}{T}), & if\ c(o) = Cyclic \\ m + rs + gt, & if\ c(o) = Increasing \\ m + rs - gt, & if\ c(o) = Decteasing \\ m + rs + \text{kx}, & if\ c(o) = Upward \\ m + rs - kx, & if\ c(o) = Downward \end{cases} \quad 5.3$$

Where m=30 and s=2. r, x, g are unifrom random values in the range of [-3,3], [7.5,20], and [0.2,0.5] respectievly. $K(t)$ equals to 1 for $t>a$ where a is a uniform random value in the range of [20,40].

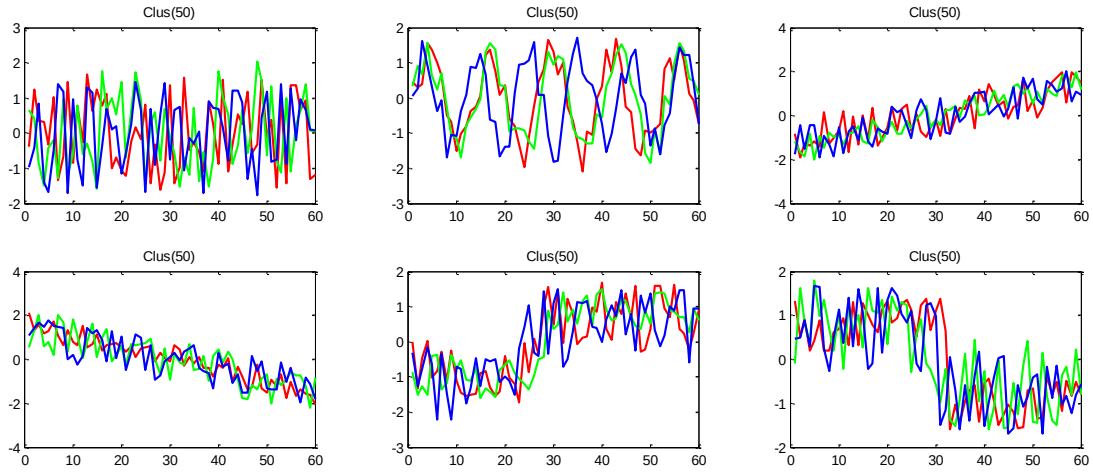


Figure 5.8: Three sample time-series of each class of CC

Moreover, different cardinalities of dataset are used in IMTC. Up to 10,000 records are created to show the experiment results.

5.3 Analyses of Methods

Generally, in the clustering of time-series data (especially for the large datasets), there are some factors which are important including database type, clustering algorithm, distance measurement and dimensionality reduction approach (representation method). For example, the type of time-series in some datasets may vary as long, short, frequent, noisy, equal length, and etc., or different types of clustering algorithms can be used such as partitioning, density based, graph based, and etc. Additionally many dimensionality reduction approaches and distance measurements have been developed which can be applied in order to reduce the dimensionality of data and calculate the similarity of time-series. Moreover, for each factor, there are some parameters to be set, which can affect the final results of clustering, for example, defining the value of thresholds, number of iterations, alphabet size, segments, and etc. As explained in research methodology (chapter 3.0), in each step of MTC also, different methods are utilized. Therefore, in the following subsections, various algorithms, dimensionality reduction approaches and distance measures used in each step of MTC are evaluated and discussed using calculating of the quality of clustering across different datasets. All methods are analysed step by step.

5.3.1 Step 1: Pre-clustering (Approximate Clustering)

In the first step of MTC, SAX is used as representation. Then, a new distance measure (i.e., APXDIST) is introduced to calculate the distance among dimensionality reduced time-series with Ek-Modes algorithm to perform clustering. In the following subsections, all these three approaches are discussed.

5.3.1.1 Representation Method (SAX)

Ratanamahatana (2005) had discussed about the quality of time-series clustering and stated that representation method is a significant component in the accuracy of clustering. Hence, it is shown how dimensionality reduction of time-series affects the

quality, by comparing the quality of algorithms with and without dimensionality reduction. The following section indicates the problem of overlooking data by showing the low accuracy of systems which use dimensionality reduced time-series and conventional clustering algorithms. Moreover, the impact of dimensionality reduction on accuracy of clustering of time-series data is described by following experiments.

5.3.1.1.1 Overlooking of Data in Clustering

As mentioned in Section 1.2, once a dataset is very large, the complexity is very high and data cannot fit in the main memory. In this case, sampling or dimension reduction is a common solution. That is, researchers use reduction, sometimes reduction to a very low resolution, in order to overcome this problem. However, the cost of sampling and data reduction is high because of missing out the data. In other words, it leads to overlooking of data, and as a result, decreases the quality of clustering.

In a scenario, let us take a dataset which consists of the time-series which represents the average of balance of customers of a bank in each day. The annual or even monthly clustering of customers based on their transactions, reveals similar customers (clusters of the same customers). Now, if two customers have the same monthly or yearly balances, one may put these customers into the same cluster. However, the value of transactions in each day or the variation of transactions during the day of these two customers may be very dissimilar. Even, one of daily transactions may consist of fraud or money laundries which are not discovered by approximating time-series data. As a result, overlooking the data may lead to loosing valuable information in data. Here, two examples of missing data are shown in the process of representation.

Example 1: Figure 5.9 depicts a special case where most of the peaks in a time-series are lost during SAX representation. Note that, SAX representation is one of the best existing representation methods, as explained in the literature review (Section 2.4.4); however, this event is unavoidable.

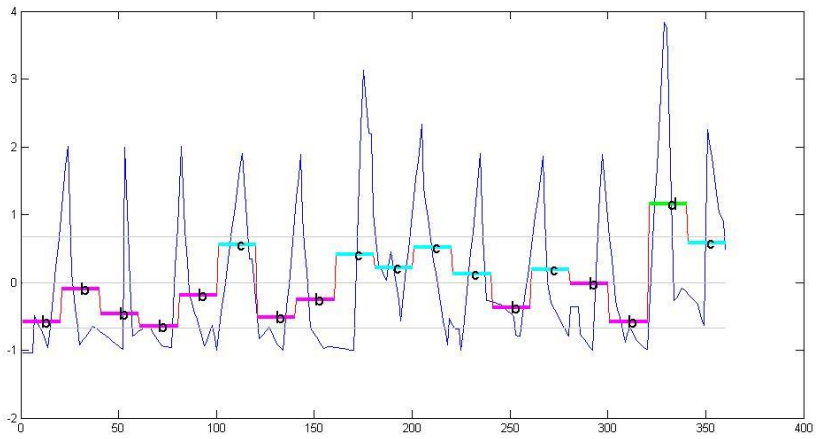


Figure 5.9: Overlooking of peaks in a time-series in SAX transformation process

Example 2: Another example is related to the same representation of three different time-series of a unique cluster depicted in Figure 5.10. It clearly points out the problem of overlooking data in the representation process (while the SAX parameters are the same for all three time-series) which leads to constructing inaccurate clusters. It clarifies how dimension reduction may lead to putting different time-series in a cluster, or how it may lose many important peaks.

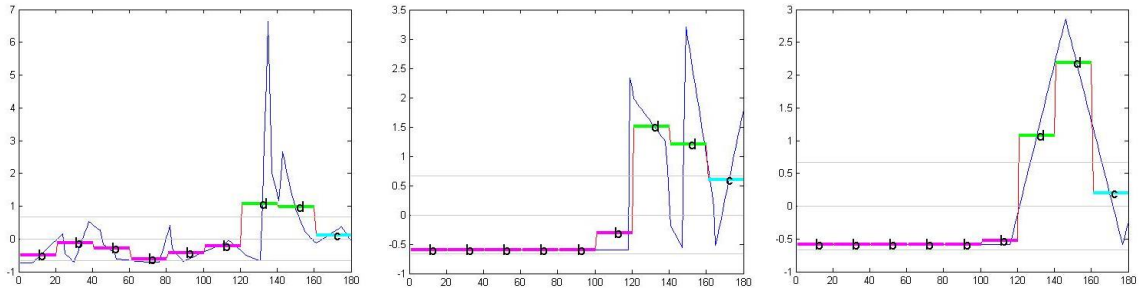


Figure 5.10: Three different time-series with similar representation method fall into a cluster

To experiment the overlooking data, the average of the qualities of SAX in front of raw time-series using different algorithms are calculated. Figure 5.11 shows how dimensionality reduction affects the quality.

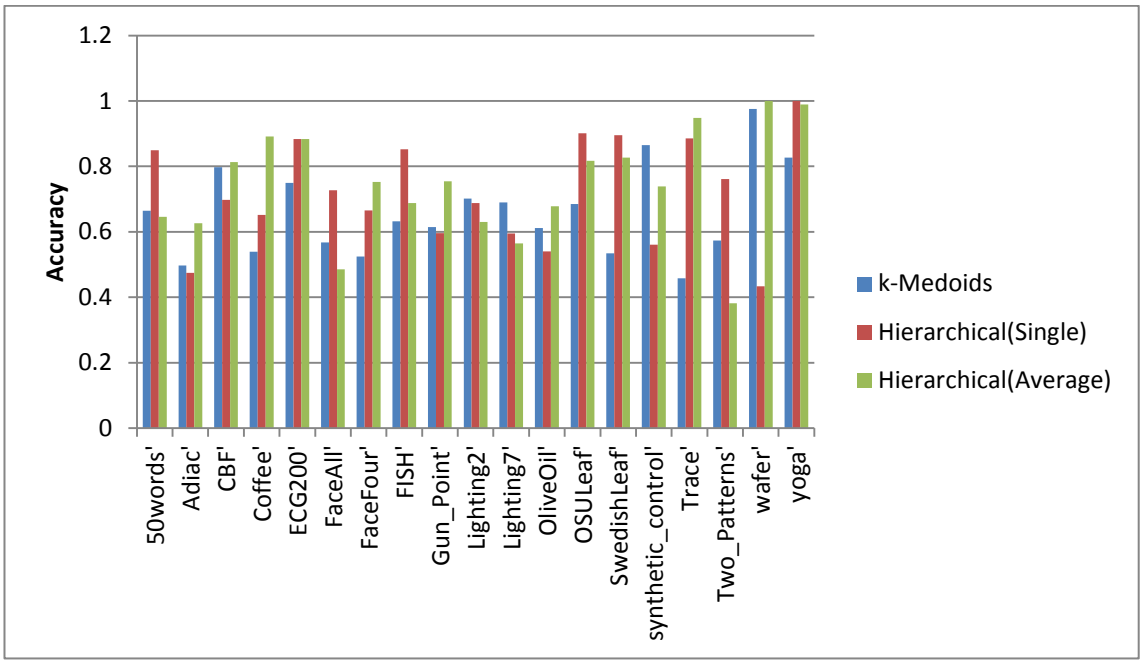


Figure 5.11: Quality of clustering of time-series represented by SAX across the raw time-series data

The quality of algorithms shown in Figure 5.11 is based on this assumption that the highest resolution of time-series generates the best answer. Although, this chart does not imply that the quality of clustering using raw time-series is better than dimensionality reduced data, it shows how different it can be. That is, as this diagram shows, if the raw data is considered as the highest resolution of the data, then the quality decreases utilizing one of the best approaches for representing data, i.e., SAX. As it is observed in Figure 5.11 the average accuracy of all algorithms is around 70%. Therefore, it can be concluded that quality of clusters is decreased around 30% by average where SAX is used as representation in comparison with using raw data.

To make it more intuitive, the average quality of clustering on all datasets in front of ground truth is depicted in Figure 5.12. In this experiment, MINDIST (J. Lin et al., 2007) with different compression-ratio values of SAX representation are considered,

i.e., SAX4, SAX6, and SAX8 (See Section 5.3.1.1.2 for more details about compression-ratio parameter).

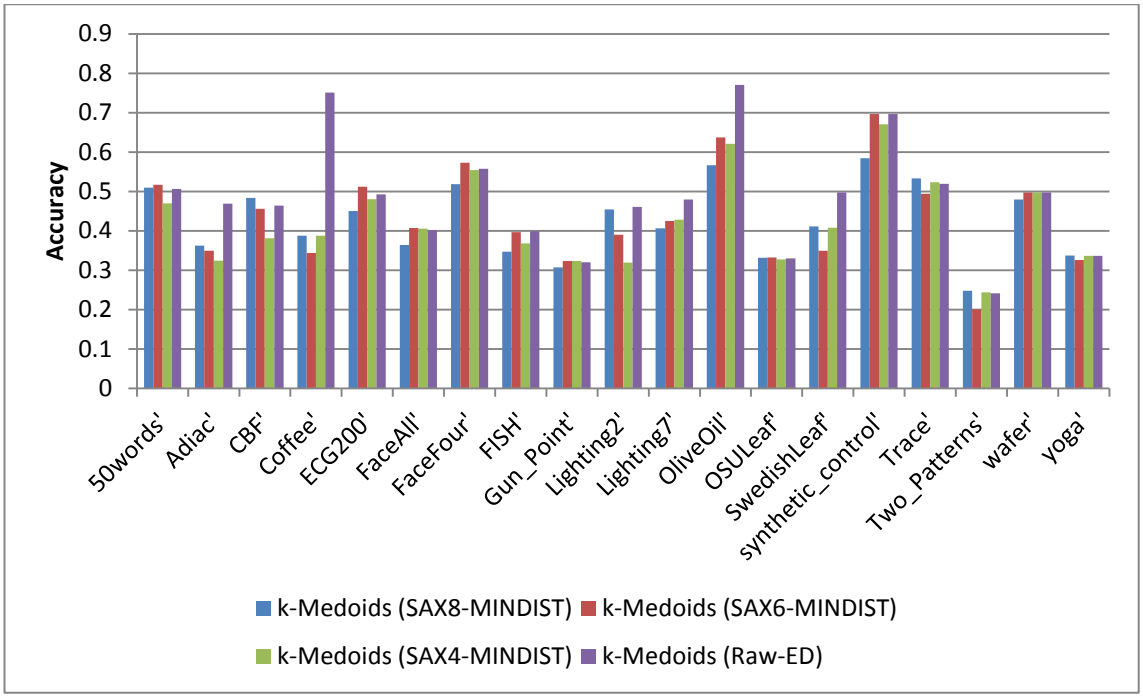


Figure 5.12: Average quality of SAX representation data in front of raw time-series across all datasets

As the above result shows, first, using a dimensionality reduction method, usually the quality of clustering is decreased (in comparison with raw time-series) for many datasets (e.g., Adiac, Coffee and Sweedish leaf). Second, using low resolution time-series does not necessarily decrease the quality for all of the datasets (e.g., Lighting2 and CBF). Instead, the quality increased in some datasets because of handling shift, outlier and noise issues in the process of representation as also was mentioned in other studies (Bagnall & Janacek, 2005). Finally, different compression-ratios of SAX, generates various qualities of clusters from each dataset. In some datasets, with increasing compression-ratios, the accuracy of the clustering improves (e.g., CBF), and in another gets worse (e.g., Sweedish Leaf). Therefore, it can be concluded that the result of clustering of approximated time-series is not precise (or at least is not reliable); sometimes it leads to worse accuracy (which is expected in most representation methods) and occasionally better quality.

5.3.1.1.2 SAX Parameter Setting

One of the problems in the representation of data using different approaches is determining their input parameters. Generally, removing parameters from a data mining task is often a good thing (Keogh, Lonardi, et al., 2004). As mentioned, in the first step of the proposed model (MTC), SAX is used as a representation method. However, SAX representation also confronts this issue to some extent because it has some parameters to be set up. In the following sections, the effect of its parameters, i.e., *word size* and *alphabet size*, on clustering quality is investigated.

SAX tries to attack the problem of high dimensionality from two different sides. It approximates a time-series in terms of length of sequence by introducing *segments* (*word size*) and in terms of amplitude by defining the alphabet size. Hence, it needs two parameters, one for controlling the granularity (word size or segments) and another one for approximating elements (Alphabet size). Determining a good trade-off is not possible for the parameters of SAX, because of its dependency to the dataset. Accordingly, (Keogh, Lonardi, et al. (2004) suggest an empirical approach to determine the best values for these parameters.

Alphabet size (denoted as 'a') is discussed in some studies (J. Lin, Keogh, Lonardi, et al., 2003; J. Lin et al., 2007) where authors try to find the best distance measure based on the tightness to Euclidean distance. The experimental result shows that alphabet sizes are not very critical (which is the strength of SAX), however they suggest choosing an alphabet size in the range of 5 to 8.

Word size or segments, as mentioned, is another SAX parameter. This parameter is indicated as compression ratio of representation which gives the user this chance to attain the ideal compression/fidelity trade-off for their particular application. Given a time-series $F_i = \{f_1, .., f_t, .., f_T\}$, the compression ratio is calculated by:

$$\text{compression}_{\text{ratio}} = \frac{T}{nSeg} \quad 5.4$$

where T is the length of time-series and $nSeg$ indicates the number of segments. In this study, different compression-ratios are used from 2 to 8. For the sake of simplicity, hereafter, different compression-ratios are shown as SAX2, SAX3,..,and SAX8.

To find the effect of compression-ratio on clustering accuracy, the quality of clusters with different compression-ratio of time-series is investigated. For this experiment, all the datasets in UCR are adopted, then, conventional k-Medoid algorithm is applied on data with various resolutions, from very low to the highest resolution (raw time-series). The quality of solution in front of ground truth is shown in the following chart (Figure 5.13).

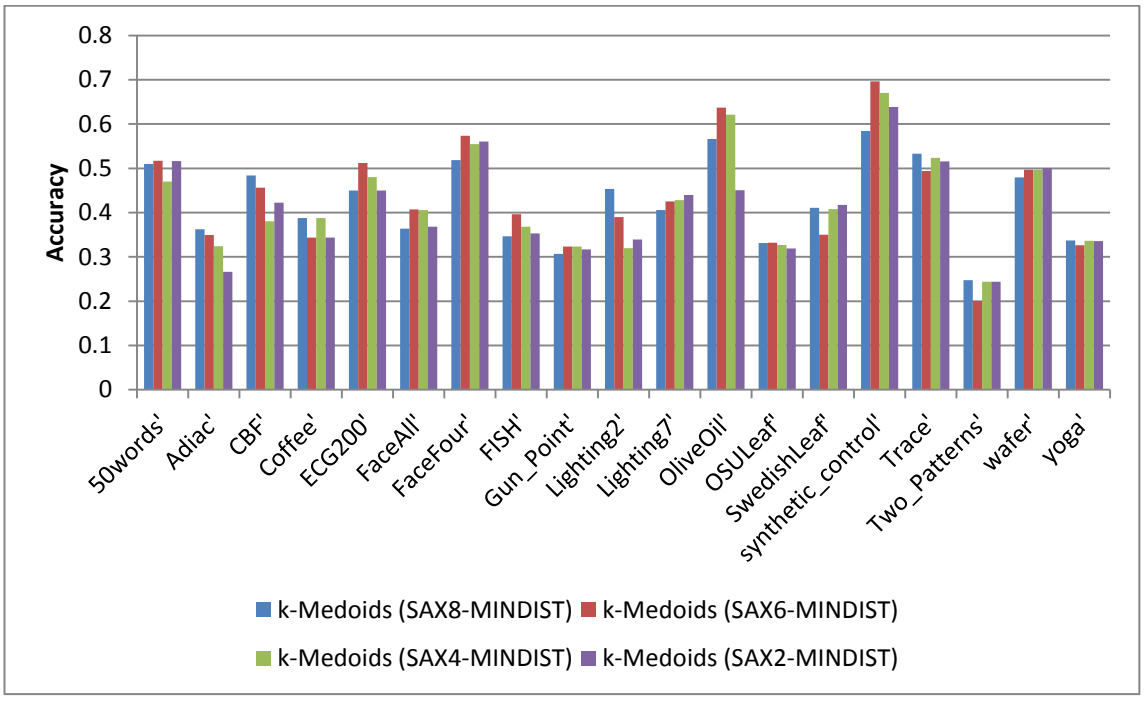


Figure 5.13: Quality of k-Medoids using SAX in front of ground truth

Surprisingly, better results are seen in higher compression-ratio (lower segments) for some datasets, but not as general. For the datasets which their higher compression leads to higher quality, it can be concluded that sometimes there are some noises and outliers in time-series which impact on the distance measures and decrease the accuracy. They

are mitigated by representation process. It is the main reason that for some datasets, lower resolutions, turns to better quality.

Furthermore, the experiment is repeated for other algorithms as well to support the validity of the experiment. In the following charts, namely Figure 5.14 and Figure 5.15 the quality of clustering algorithm on different resolutions of time-series are investigated, and the results are reported in front of ground truth.

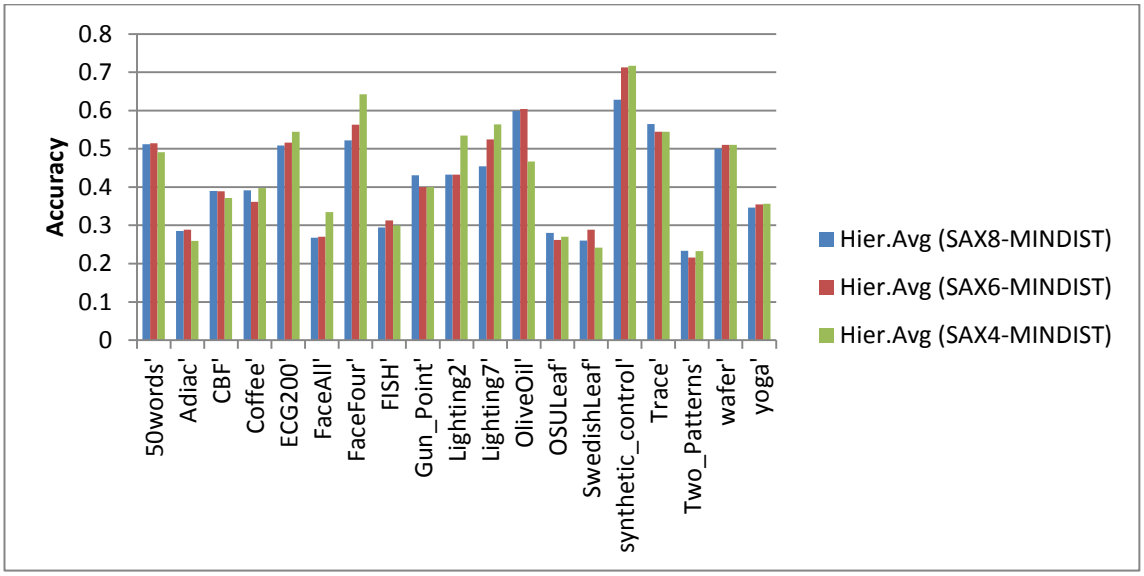


Figure 5.14: Quality of hierarchical (Average linkage) clustering of time-series representing by SAX in front of ground truth

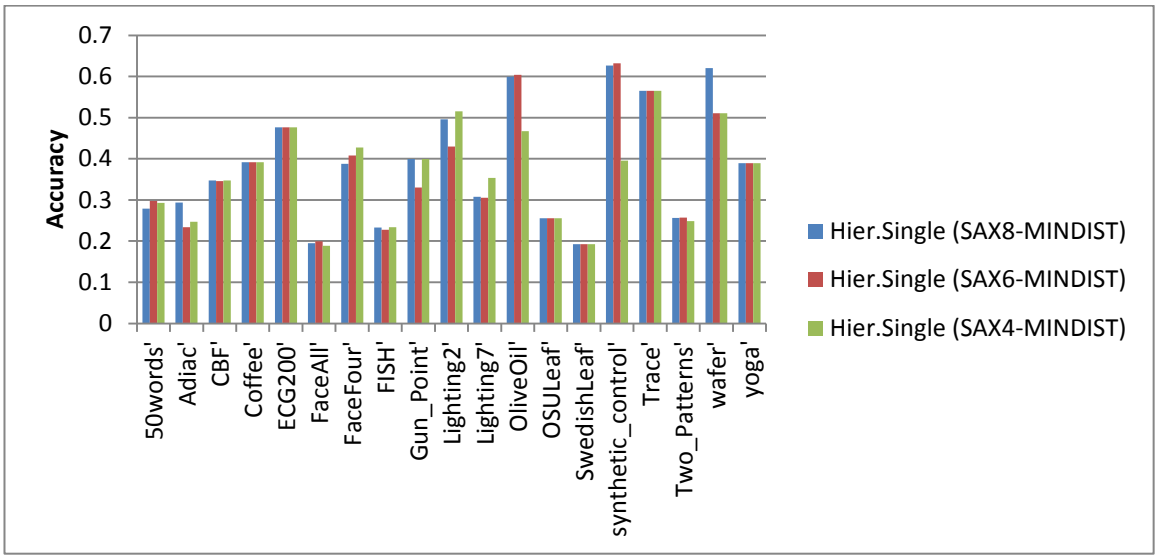


Figure 5.15: Quality of hierarchical (Single linkage) clustering on time-series representing by SAX in front of ground truth

As depicted in Figure 5.14 and Figure 5.15, in most of the algorithms, there are various changes in different compression-ratios. That is, with increasing of resolution of time-series, the accuracy of clustering changes around 10% for some datasets. It means that compression-ratio of SAX affects the quality expectedly, and it is not very stable in front of different ratios.

5.3.1.2 Distance Metric (APXDIST)

The distance metric which is used to calculate the similarity between time-series is a very important and essential part of time-series clustering. As mentioned in the methodology chapter (chapter 3.0), the distance measure designed for this step is APXDIST (see 4.3.3 for details of design). This distance measure was suggested (for answering to the first research question of this study) in order to improve the accuracy of similarity measurement between approximated time-series. In this section, APXDIST measure is evaluated.

In the first experiment, the UCR datasets are again considered to test the new distance measure, APXDIST, in front of MINDIST and ED. MINDIST is the measure which is compatible with SAX and was suggested for indexing purpose (J. Lin et al., 2007). ED is the measure which is used in related works (Lai et al., 2010) for calculating the Euclidean distance between two time-series representing by SAX. To make a comparison, at first, the dataset is transferred into discrete space (SAX). Then, three distance matrices are made by each distance measure (i.e., APXDIST, MINDIST and ED). At that point, the difference between the obtained distance matrices and the distance matrix calculated by Euclidean distance (using raw time-series) is computed as tightness of each metric to Euclidean distance (ED). The tightness of the metrics, for example APXDIST, to Euclidean distance is calculated by the following equation.

$$Tightness(\hat{F}_i, \hat{F}_j) = 1 - \frac{|ED(F_i, F_j) - APXDIST\ (\hat{F}_i, \hat{F}_j)|}{ED(F_i, F_j)} \quad 5.5$$

where $\hat{F}_i$ is the SAX representation of time-series F_i. The average tightness of all UCR datasets over the cross product of the alphabet [5-8] and compression-ratio [4-8] is shown in Figure 5.16. The bigger tightness indicates more accurate approximation.

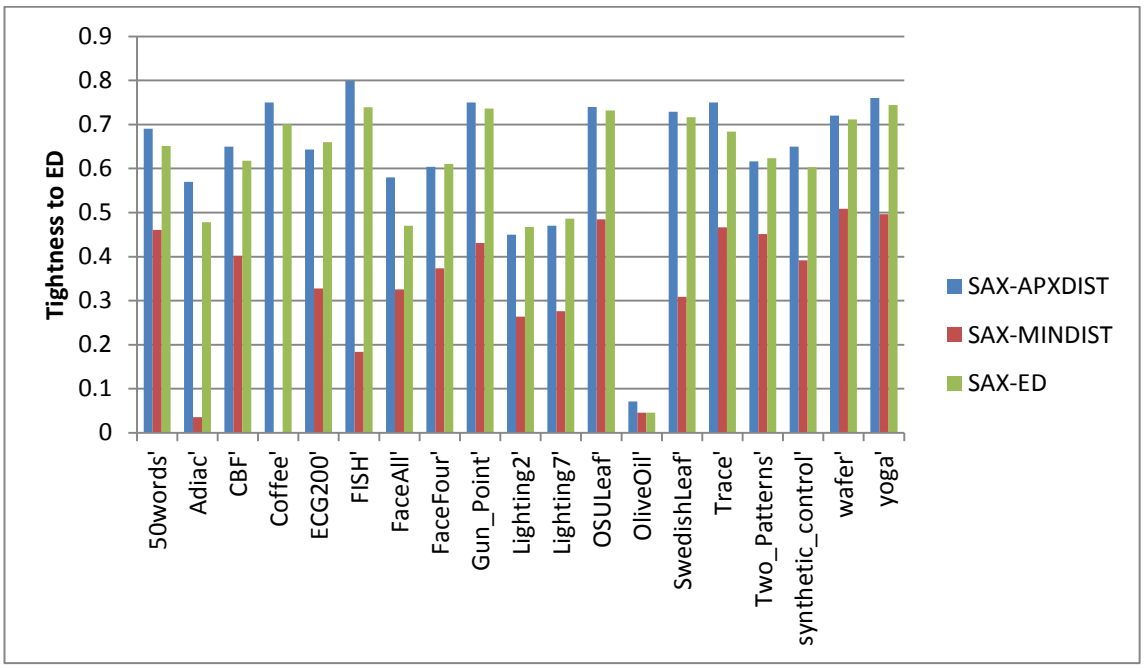


Figure 5.16: The tightness of APXDIST to the Euclidean distance

As the results show, APXDIST is closer to Euclidean distance (on raw time-series) rather than the other two approaches for most of the datasets. However, in some cases, Euclidean distance works better on time-series represented by SAX (e.g., Lighting7). These cases, can occur once the time-series in a dataset has a high frequency with many peaks. As a result, it may cause a decrease in normalized distribution property between the symbols which is the pre-assumption in SAX parameters.

Moreover, to show the behaviour of different distance metrics in front of ground truth, they are tested in front of ground truth. In this experiment, SAX representation (SAX6) is used against different clustering algorithms (k-Means, hierarchical with average linkage, hierarchical with single linkage, k-Modes and k-Medoids). The average quality of clustering in front of the ground truth is shown in Figure 5.17.

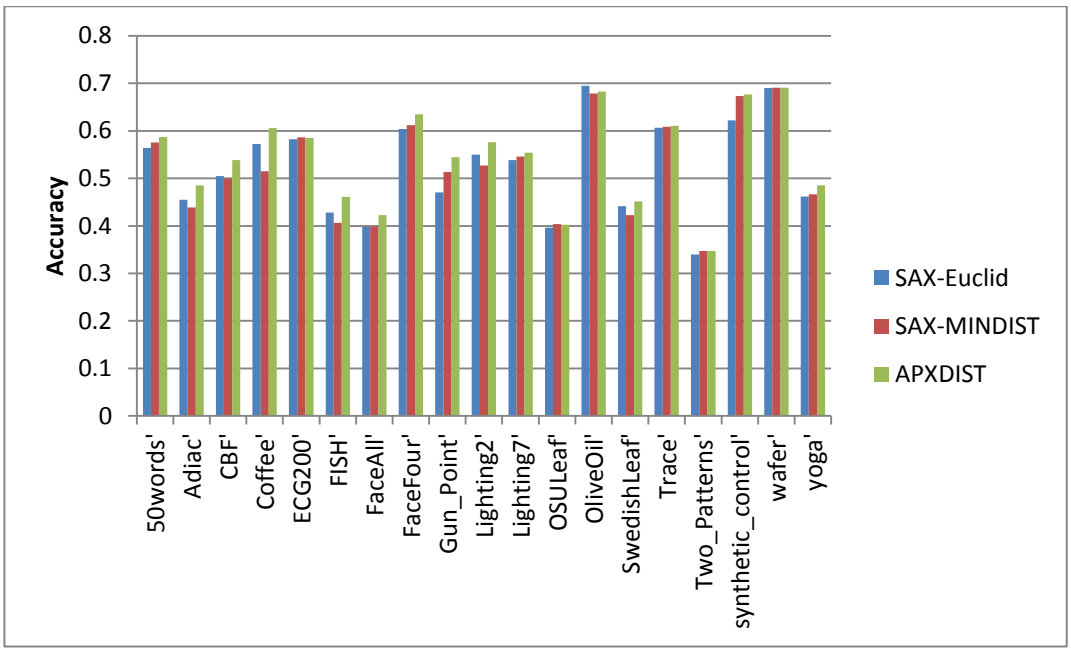


Figure 5.17: The average quality of clustering in front of the ground truth

Based on the results obtained on different datasets, it is obvious that APXDIST works better than other approaches in most of the datasets. For example, in synthetic datasets, MINDIST works better than ED. Similarly, APXDIST also works fine. However, in the case where ED is working better, such as on Coffee dataset, APXDIST also works better than MINDIST. As a result, it can be concluded that in general, APXDIST is more stable than others exploiting the strengths of ED and MINDIST. Although, the effect of these distance measures on quality of clustering is not very high, it can be important in large datasets (high cardinality datasets) and sensitive datasets. Moreover, in the next section it is shown how APXDIST can improve the quality using an appropriate clustering algorithm.

5.3.1.3 Algorithm (Ek-modes)

The proposed algorithm for symbolic representation is the Ek-Modes explained in Section 4.3.4. Here, in the first attempt, the quality of clustering in front of different algorithms is calculated to investigate the influence of different algorithms on datasets, which is also the answer to the second research question of this study. The quality of clustering are measured in this experiment using APXDIST in front of k-Means, k-

Mediods, and hierarchy algorithms. In this experiment, SAX8, SAX6 and SAX4 are used as representation methods, and APXDIST has been used as a distance measure. The average of 100 times run is shown in Figure 5.18.

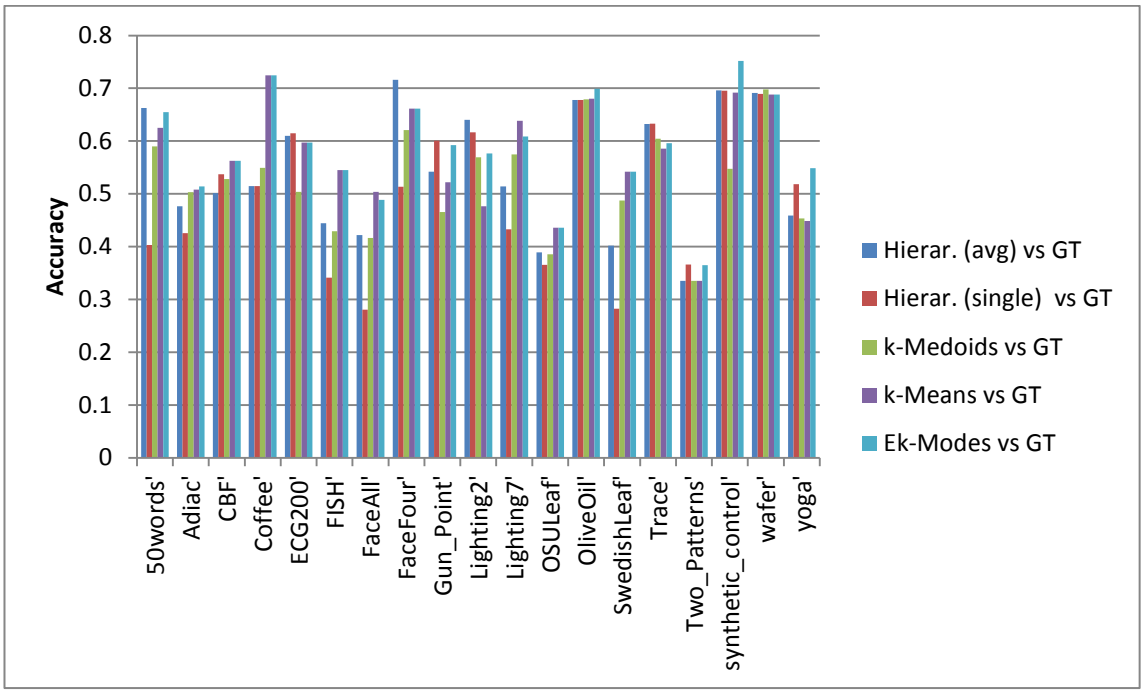


Figure 5.18: Average quality of clustering of time-series represented by SAX representation vs. ground truth (GT) on UCR dataset (TRAIN set)

The result reveals that generally Ek-Modes algorithm has relatively better results than other algorithms, and it is a good choice as a superior algorithm for pre-clustering. However, the average quality is calculated and is depicted graphically by a box plot as depicted in Figure 5.19.

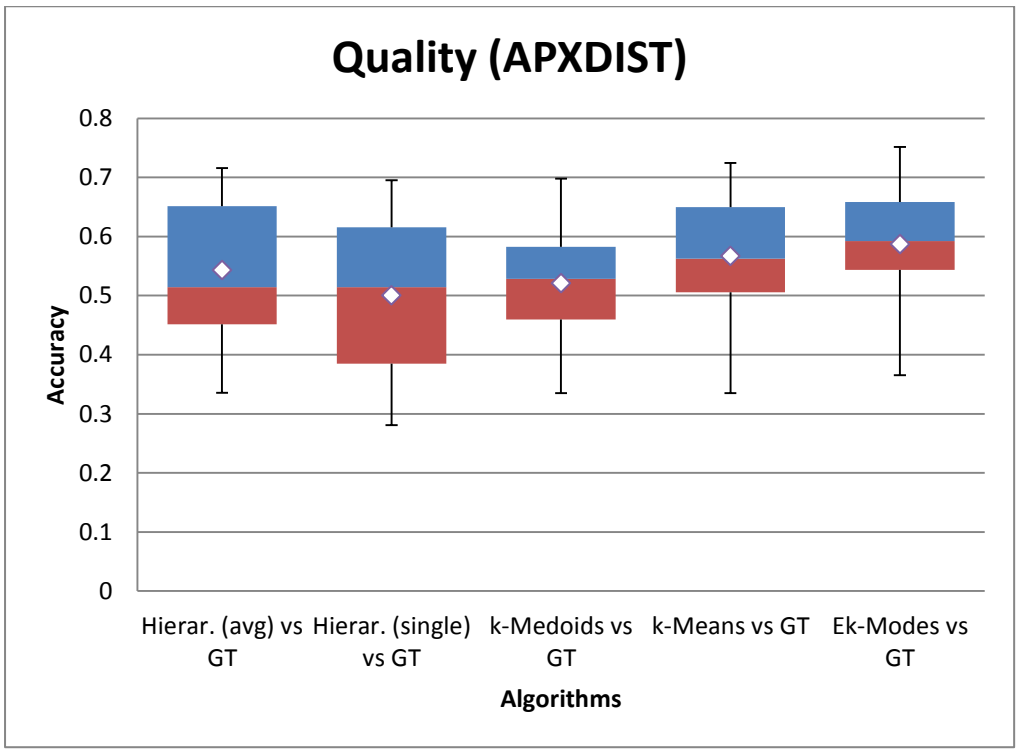


Figure 5.19: Quality of different algorithms in the first step (across GT)

As depicted in Figure 5.19, the average quality shows different accuracies for all algorithms against ground truth. The experiment shows that the quality of approximate clustering by hierarchical and partitioning algorithms is relatively the same against ground truth. However, totally Ek-Modes clustering has better results when the dimension of time-series is reduced. Considering this concept that Ek-Modes algorithm is of type of partitioning algorithm, it exploits from high speed (Huang, 1998) in comparison to other algorithms such as hierarchical algorithms. Hence, the best choice for the approximation clustering (i.e., generating pre-clusters) in the first step of MTC is the Ek-Modes algorithm.

5.3.2 Step 2: Purifying and Summarization

The objective of second step of MTC is revising the pre-clusters by splitting them into sub-clusters. The output of this step is some pure sub-clusters represented by prototypes. This action needs high (higher in comparison with first step) resolution of time-series and an accurate (or more accurate in comparison to the first step) distance measures. As explained in previous chapter, in this chapter, the concern is generating

less but purer sub-clusters, that is, more reduction and less error rate (see Section 4.4.2.2). In the following subsection, the purity and the amount of reduction of data are evaluated.

5.3.2.1 Reduction vs. Purity

As mentioned, the objective of this step is to obtain the purest sub-clusters and the least number of sub-clusters. The purest cluster is the one where its members belong to the same classes of ground truth. To achieve to this objective, PCS was designed in Section 4.4.2. The purity of a sub-cluster depends on the value of affinity threshold, α, in the PCS algorithm (in the case that it is not dynamic). The higher α is, the purer sub-clusters are gained. In contrast, finding inaccurate sub-clusters leads to inaccuracy in the final results. However, the cost of purer sub-clusters is the higher number of sub-clusters. Of course, the sub-clusters with only one time-series are the purest clusters. However, the cost of merging a lot of time-series in the last step is expensive. Hence, it needs a trade-off between the purity and number of sub-clusters. The following experiments verify that the proposed method in Section 4.4.2, i.e., PCS, can reduce the size of data but not reduce its effectiveness greatly.

A parameter is defined as reduction-rate to find the best α. As mentioned, in the proposed model (MTC), each pre-cluster is broken down into purer sub-clusters. Given k_{CS} as the number of sub-clusters, then, the reduction-rate of second step is defined as:

$$R_{rate} = 1 - k_{SC}/\mathrm{n} \qquad 5.6$$

where R_{rate} is reduction-rate and k_{CS}/n is ratio of the size of sub-clusters to the size of dataset. Different values of α are used (in the second step of MTC model) to show how reduction-rate changes. For example in front of α =0.7, the result is depicted in Figure 5.20.

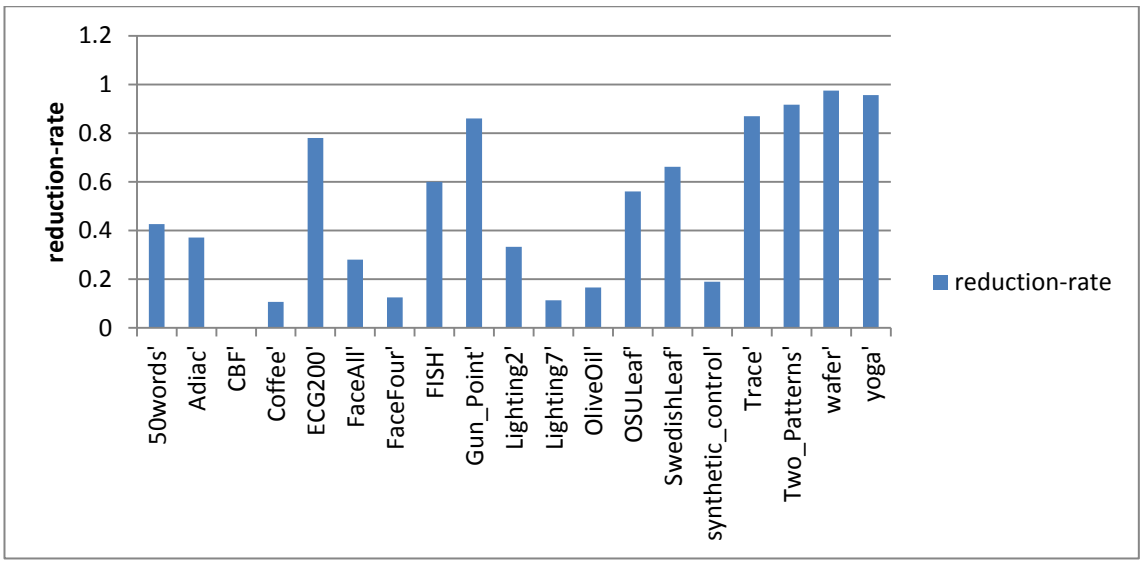


Figure 5.20: Reduction-rate in the second step of MTC against the number of clusters for α =0.7

Figure 5.20 reveals that for some datasets, very good reduction is gained, e.g., Wafer, but not for all. How much purity is obtained per reduction of data? Corresponding to reduction-rate in the second step, the purity of sub-clusters against the ground truth is calculated. Here, purity of sub-clusters are calculated based on the number of items in the same sub-cluster that belong to the same class (ground truth) (Van Rijsbergen, 1979) as it was discussed in Section 2.7.1. For example, in the case that α =0.7 was chosen as affinity threshold for PCS, then the purity of first and second steps are depicted in Figure 5.21.

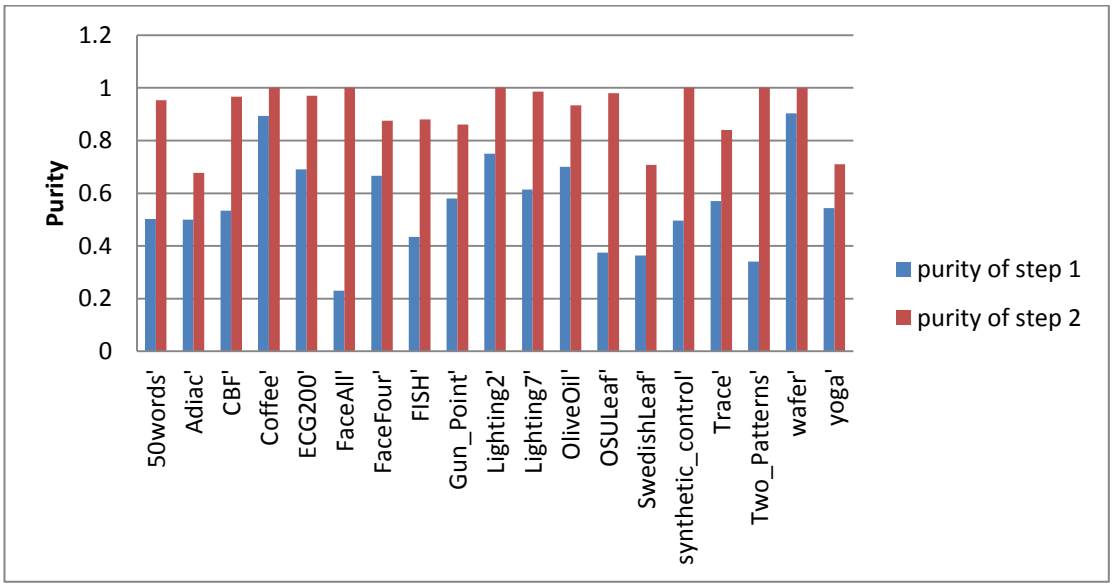


Figure 5.21: Purity of first step and second step for α =0.7

The results in Figure 5.21 show the purity of different datasets in the first and second step clustering by MTC using $\alpha = 0.7$. As an example, for the first dataset (50words dataset), purity of the obtained clusters in the first step is around 50%, but the purity of sub-clusters in the second step is increased by around 90%. However, it should be noticed that the number of clusters in the second step is more than first step which is around 60% of the whole number of time-series in dataset (see Figure 5.20). In Figure 5.22, the proportion of reduction-rate and purity of sub-clusters in the second step is illustrated using $\alpha = 0.7$.

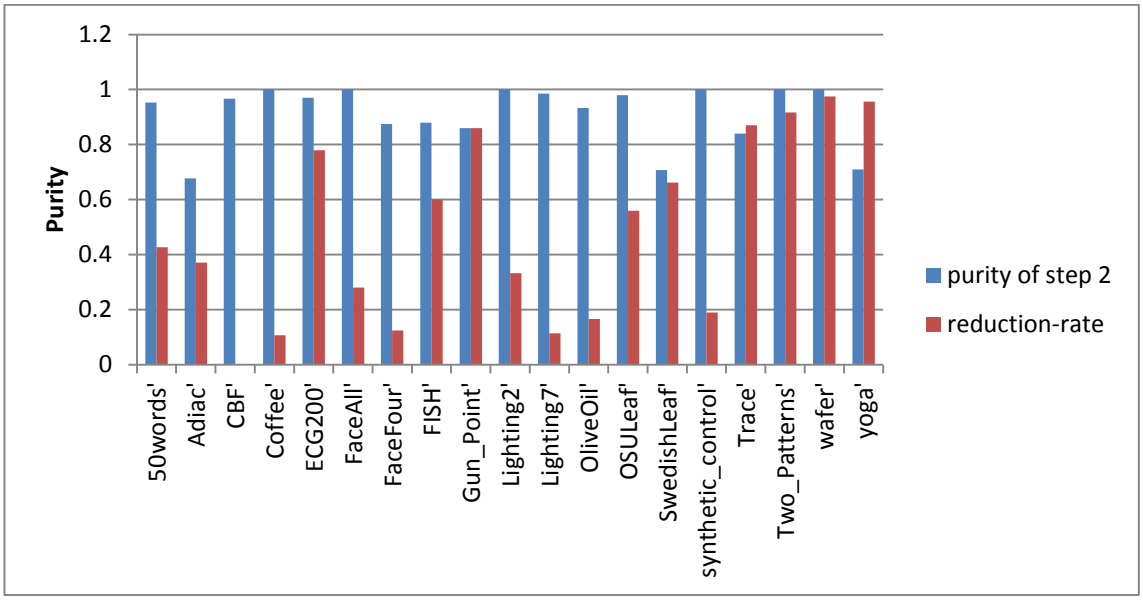


Figure 5.22: The reduction-rate and purity of the second step for α=0.7

Figure 5.22 reveals that in some of the datasets, the effect of reduction on purity is very high (e.g., Adiace dataset), and in some of the datasets is not more than 10% (e.g., 50words dataset). Therefore, it can be concluded that raising the purity of the second step clusters, increases the number of sub-clusters, but with different ratios. The results in this chart is related to $\alpha = 0.7$, which increases the purity of pre-clusters very much (in comparison with the first stage) but it reduces the number of instances a bit. In MTC, a new dynamic α was introduced for PCS algorithm which can find a good trade-off between reduction-rate and purity using the approximate affinity between time-series in pre-clusters (see Section 4.4.2.2 for more details). Therefore, the experiment

should be repeated for some α to show the effectiveness of the proposed dynamic α, so called $\alpha = \text{PCS}$.

To find a good trade-off between the reduction-ratio (R_{rate}) and purity of sub-clusters (P_{SC}), the objective function is defined as the quality-gain-ratio:

$$\text{QGR} = \frac{P_{CS} + R_{rate}}{2} \qquad 5.7$$

In order to show the effectiveness of α in PCS, the experiment is carried out for $\alpha = [0.3, 0.7]$ across all datasets. The result is shown in Figure 5.23.

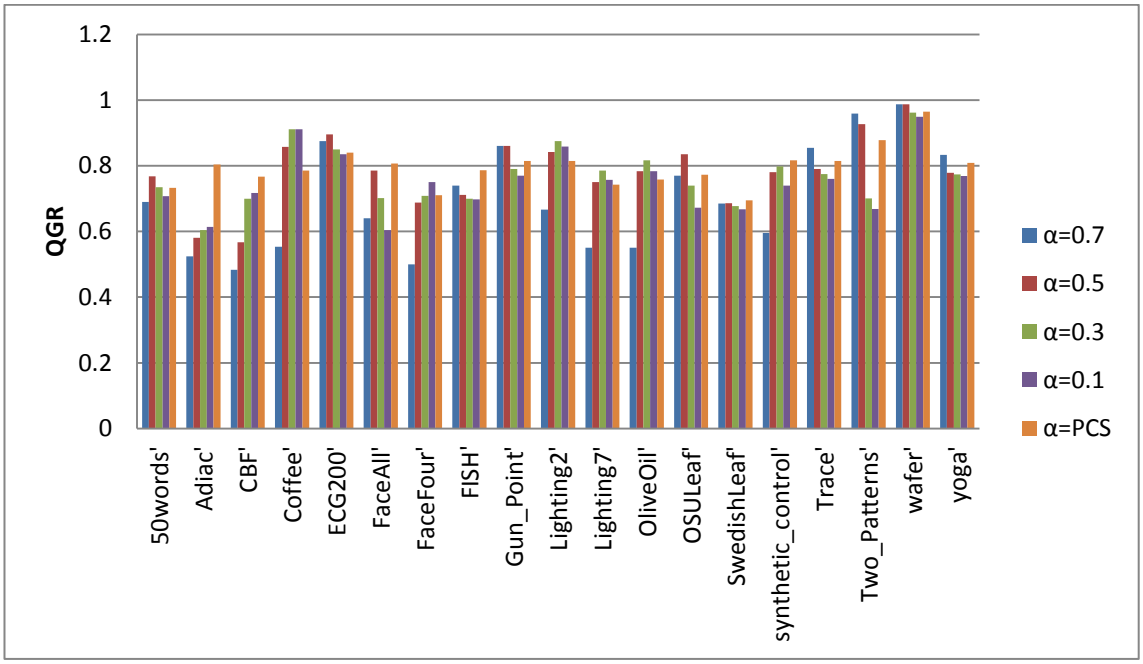


Figure 5.23: The QGR of second step per different values of α

This chart shows that $\alpha = \text{PCS}$ is outperforming (by average) in comparison with static values of α, providing a good trade-off between quality and reduction-rate, and at the same time, it is dynamic. The dynamic characteristic of PCS is very important because it gives a high QGR for most of datasets but other static thresholds gives a high QGR only for a few datasets (e.g. $\alpha = 0.7$ is very good on Yoga and Two pattern, but it is failed for other datasets). However, if the high quality is desirable in a specific dataset, higher α (defined by user) can result in more accurate clusters. Figure 5.24 shows the

QGR for α = PCS and the proportion of the purity and reduction-rate for different datasets.

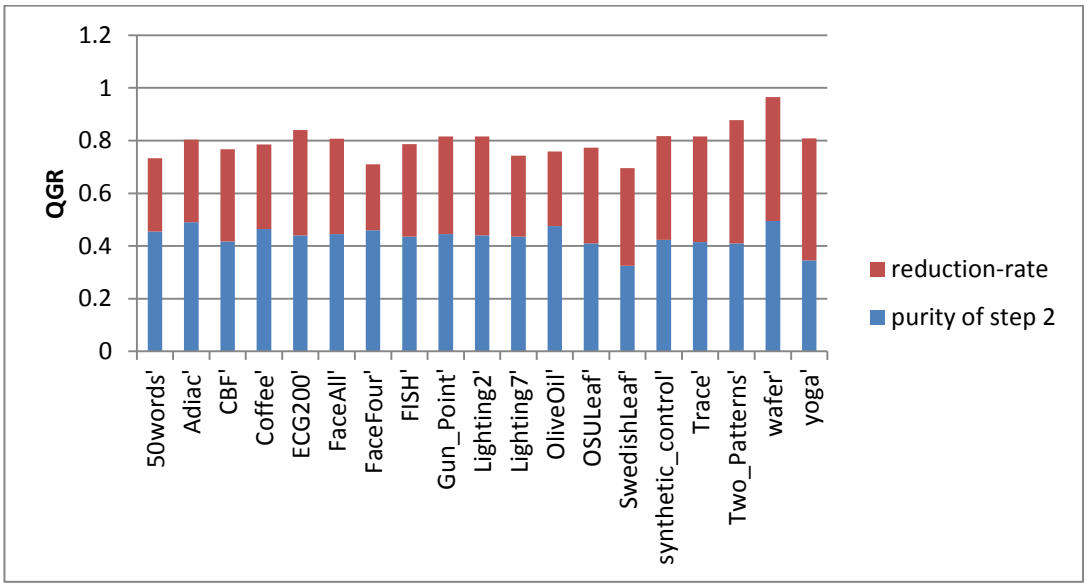


Figure 5.24: PCS approach reduction-rate in front of gained purity

Figure 5.24 shows QGR of PCS using dynamic α. As the chart reveals, a good trade-off between reduction-rate and purity is obtained without parameter settings. That is, in PCS algorithm, the affinity threshold is adjusted dynamically regardless of characteristics of time-series or size of data which is of great important. To show how different α can affect the purity and reduction-rate of second step clusters, the experiment is shown specifically for different α on CBF dataset as depicted in Figure 5.25.

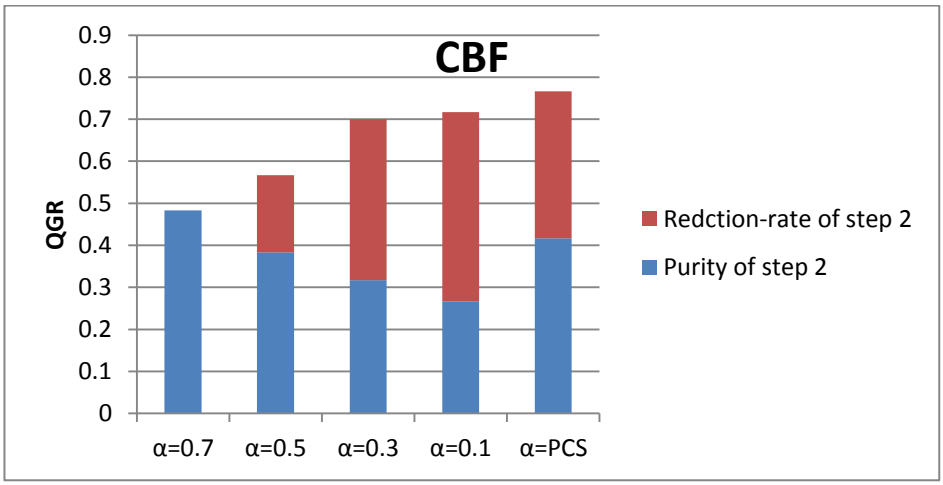


Figure 5.25: The proportion of reduction-rate in front of quality for different values of α

As Figure 5.25 shows, PCS provides better QGR in comparison with other α values. That is, using this approach, a good trade-off between quality and reduction-rate is gained dynamically. All these experiments were carried out on TRAIN set of UCR dataset which is not very large. However, the effect of PCS is more obvious on large datasets such as Wafer dataset. Hence, the experiment was repeated on CBF with different cardinalities to show the effect of PCS on large time-series. As expected, the PCS works very well on large datasets as it is depicted in Appendix F.

Therefore, using a post-clustering algorithm, pre-clusters are revised in such a way that they are broken into purer clusters and then they are represented by prototypes. It is the answer to third research question of this study. Moreover, comparing prototypes with the original data, it is understood that using this approach, the data size is reduced dynamically based on the characteristics of time-series.

5.3.3 Step 3: Merging

As mentioned in Section 4.5, the objective of the third level of MTC is merging the sub-clusters represented by their prototypes (representatives). In the following section, the distance metric used for calculating the distance between prototypes is evaluated, and then the utilized scheme for merging are discussed.

5.3.3.1 Distance Measurement

Prototypes of sub-clusters are high-resolution time-series which precisely represent the sub-clusters. As explained in Section 4.5.1, for similarity calculation between two prototype (which are high resolution time-series), DTW is used as an accurate distance measurement. At first, the quality of clustering of different datasets using ED and DTW is investigated as two of the most-used distance measures in the literature (as discussed in Section 2.5.1).

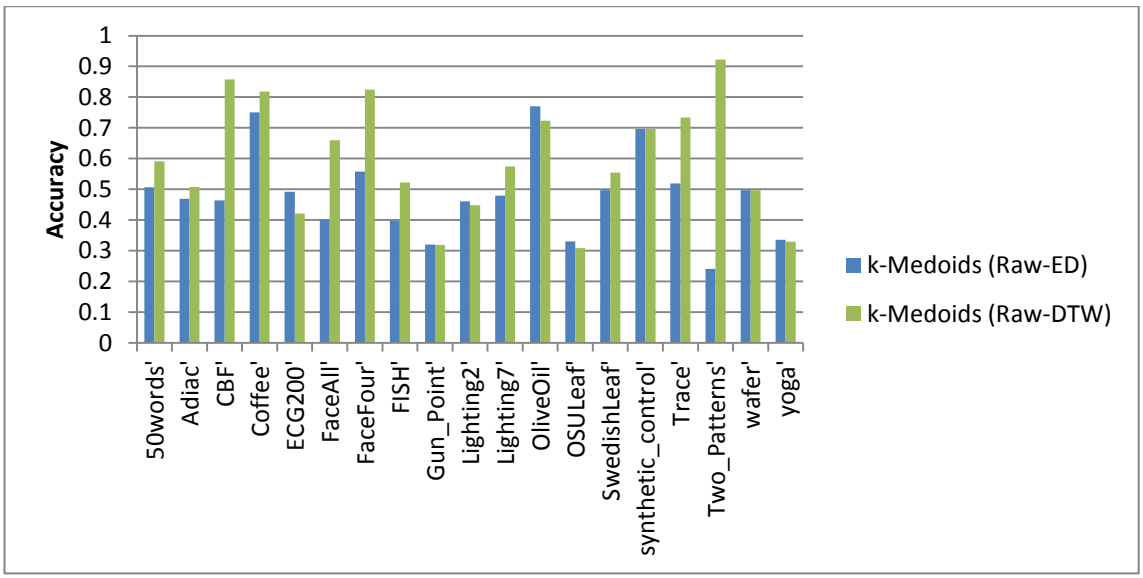


Figure 5.26: Quality of clustering in front of ED and DTW

Considering the results shown in Figure 5.26, expectedly, DTW is a better choice in comparison with ED in terms of quality because of handling shifts in calculating the similarity. The experiment shows that it is true in large datasets (see Appendix D). This improvement in quality is at the expense of high complexity of using DTW. Although the number of prototypes are small in comparison with original data, DTW can be used with some constraints (warping window) to mitigate the high complexity of distance measure as explained in Section 4.5.1. There is a range of [ED, DTW(10%), DTW(20%) , .., DTW(90%), DTW(100%)] as different warping windows from ED (as upper bound of DTW) to DTW(100%) (as complete DTW), i.e., $DTW_r = [1\ to\ d]$. In a study (Ratanamahatana & Keogh, 2005), the authors investigate the required percentage of warping window in DTW calculation for different datasets. Their results show that it is different and depends on the nature of dataset. Hence, to achieve the maximum accuracy, complete DTW should be calculated for some dataset.

5.3.3.2 Merging Scheme

As explained in research methodology, at the third step of MTC, an arbitrary clustering scheme (e.g., hierarchical or partitioning) can be used depending on the nature of the dataset. In order to experiment MTC, k-Medoids, hierarchical by single linkage,

hierarchical by average linkage, and the arbitrary shape cluster approach is implemented as merging scheme for the third step. In Section 4.5, it was shown that using multi-prototype and a customized hierarchical algorithm, merging can be performed for arbitrary shape clusters. In the following figures, two different datasets were chosen to show the effect of arbitrary shape clustering (see Figure 5.27 and Figure 5.28). In this experiment, MTC is performed using different schemes for the third step to show its performance.

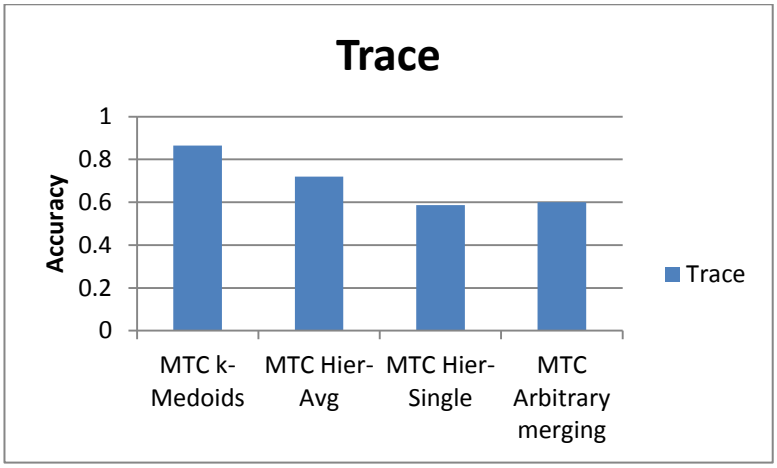


Figure 5.27: Using schemes of merging on Trace dataset

Figure 5.27 shows that if the size of clusters is equal or comparable size (e.g., Trace dataset), the clusters are globular (spherical) and the partitioning clustering is a good choice.

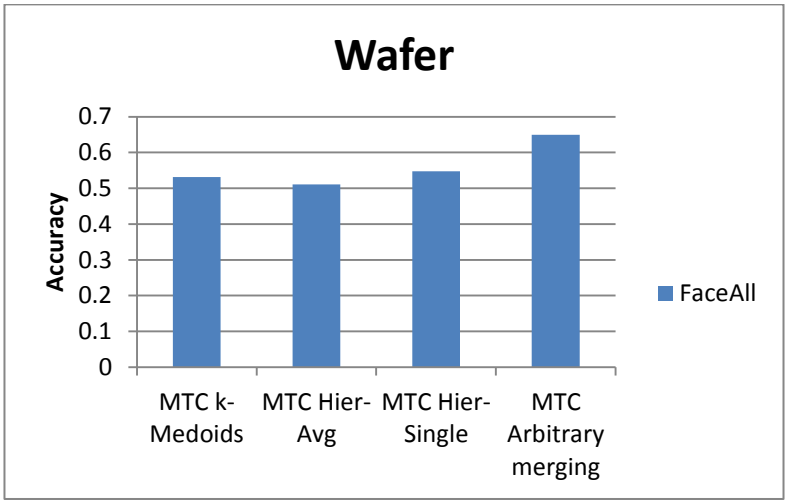


Figure 5.28: A sample of gained accuracy using MTC with the arbitrary shape scheme in the third level.

However, as Figure 5.28 shows, in the case that clusters are of different sizes, in most of cases arbitrary shape clustering outperforms. The result of running arbitrary shape clustering on all datasets is reported in Appendix I.

5.4 Final Results

In this section, MTC is applied on all datasets and the results are reported, and then, in the next chapter, they are visualized, compared and discussed. However, before running clustering algorithm, providing a fair condition, each instance of the dataset is normalized. That is, each time-series changes to have zero mean and have a standard deviation of one. Indeed, z-normalization improves substantially the clustering accuracy irrespective of the chosen distance measure. The formula for normalization is given in Section 4.3.1. To report the result of our proposed model (MTC), much iteration is performed with various parameter settings. The parameter used to report the final result is depicted in Table 5.2.

Table 5.2: Setting up of parameter values of MTC to generate the experimental results

Step	Parameter	Description	Range
1	SAX compression-ratio ($\text{compression}_{\text{ratio}}$)	The value of compression-ratio for the SAX representation method in the first step 1 of MTC	[4-8]
2	Resolution	Resolution of time-series	SAX2,Raw time-series
3	DTW warping window (DTW_r)	The DTW warping window as a constraint for speed up calculation	75%, 100%
3	Algorithm scheme	The type of clustering algorithm for the third step	Hierarchical (average, single linkage), Partitioning (k-Medoids)
3	Cluster numbers (k)	In particular, k is defined based on the number of clusters in labelled dataset (user definition).	-

In the experiment results reported in this section, the range of values (shown in Table 5.2) is used as the parameter values of MTC in applying on all datasets. A parameter

study also is performed to determine the sensitivity of MTC on the above set of parameters by using SAX's $\text{compression}_{\text{ratio}} = [4, 6, 8]$ in Section 6.4.

As mentioned, the measure of quality in this experiment is the average value of criteria discussed in Section 2.7.1. For different parameter combinations, the average quality is reported in Table 5.3. The quality reported in each experiment, is the average value of external criteria discussed in Section 3.2.7. The result and accuracy of MTC on each step are shown for some datasets in detail in the next chapter.

Table 5.3: The result of MTC in comparison with conventional algorithms

Algorithm / Dataset	k-Medoids	MTC k-Medoids	Hier (Avg)	MTC Hier (Avg)	Hier (Single)	MTC Hier (Single)
50words	0.499	0.615	0.506	0.578	0.29	0.343
Adiac	0.345	0.498	0.278	0.443	0.258	0.284
CBF	0.44	0.836	0.383	0.623	0.347	0.328
Coffee	0.373	0.628	0.384	0.453	0.392	0.422
ECG200	0.481	0.44	0.523	0.435	0.476	0.451
FaceAll	0.392	0.587	0.291	0.405	0.194	0.218
FaceFour	0.549	0.681	0.576	0.438	0.408	0.392
FISH	0.371	0.486	0.303	0.366	0.231	0.231
Gun_Point	0.318	0.363	0.41	0.42	0.377	0.388
Lighting2	0.388	0.469	0.467	0.498	0.48	0.485
Lighting7	0.42	0.598	0.514	0.49	0.322	0.349
OliveOil	0.608	0.688	0.557	0.687	0.557	0.712
OSULeaf	0.33	0.386	0.271	0.313	0.255	0.261
SwedishLeaf	0.39	0.513	0.264	0.35	0.193	0.21
synthetic_control	0.65	0.865	0.686	0.72	0.551	0.587
Trace	0.517	0.764	0.551	0.745	0.565	0.734
Two_Patterns	0.231	0.723	0.228	0.639	0.254	0.846
wafer	0.491	0.532	0.507	0.51	0.547	0.547
yoga	0.333	0.36	0.353	0.368	0.389	0.39

As Table 5.3 shows, the quality of MTC is superior for most of datasets using different schemes of clustering. The results (which are discussed further in Section 6.4) shows that MTC is not very sensitive on the mentioned parameters, and MTC can generate accurate clusters for all of these combinations of values for distance method, and merging algorithms. Moreover, the maximum accuracy for each scheme of MTC is shown in Appendix G. The results are discussed more in the next chapter by comparing with other algorithms.

5.5 Chapter Summary

In this chapter, the spectrum of datasets used for the experiment was depicted. The chosen datasets for experiment are from various domains and different sizes. In the next part of the chapter, each step of MTC was analysed. In each step, the designed method/methods in the previous chapter were evaluated separately using different datasets which are also advantageous for other researchers. Strength and scalability of SAX were explained. It was shown that while SAX is very scalable, it causes overlooking in some datasets. Then, APXDIST was evaluated by comparing with other distance measures. It was justified why APXDIST works better for dimensionality reduced time-series. For the first time, k-Modes algorithm was used on time-series represented by symbolic representation which leads to more stable results in comparison with other partitioning and hierarchical methods used in this domain. Then purifying of approximated clusters was discussed. It was shown how purifying by PCS can split a cluster into pure sub-clusters. The process of purifying was exposed numerically. It was shown that using affinity concept, the size of time-series data can be reduced without much violating the accuracy. Finally, sub-clusters were merged by various algorithms. In different runs, it was shown how a user can choose different schemes such as partitioning or hierarchical algorithm to merge the prototypes in the third step. The final

results of running all steps of MTC were reported which show its superiority in comparison with existing methods. This will be discussed more in the next chapter.

6.0 EXPERIMENTS EVALUATION

6.1 Introduction

One of the major obstacles in the data mining domain is the difficulty of evaluation of a clustering algorithm without taking this context into account: why does a user cluster his data in the first place, and what does he want to do with the clustering afterwards? Answering these questions widely depends on the domain of the problem in hand. However, as explained in the research methodology, some common approaches are taken here to evaluate the model. Essentially, the performance of a time-series clustering is evaluated with some parameters. For the proposed clustering model, MTC, the following parameters are used for evaluation:

1. Accuracy evaluation: The quality of the final clusters according to the ground truth
2. Scalability evaluation: Execution time and memory usage of algorithm
3. Sensitively evaluation: Important parameters that affect the clustering results

In this chapter the effectiveness of the proposed model (MTC) will be validated with performed experiments. In addition, the accuracy of the final results of the model (MTC) is evaluated, and compared with other widely used approaches. Although the objective of this thesis is enhancing accuracy of time-series clustering, the scalability and sensitivity of the MTC are evaluated to support the claim of the practicality of the method. It is shown that the proposed approach (MTC) leads to clustering of time-series effectively and efficiently as well.

6.2 Accuracy Evaluation

One of the challenges in time-series clustering is evaluating the quality of clustering results which is not a trivial task. It is mostly due to its unsupervised learning manner because of absence of ground truth (actual clusters). Whether the ground truth exists or

not, **Visualization** and **Scalar measurement** methods are two major techniques for evaluation of clustering quality (also is known as clustering validity) (Hathaway & Bezdek, 2003). In this study both methods are used to evaluate the accuracy of MTC in the following sections.

6.2.1 MTC Visualization

In this section, all three steps of MTC are depicted using two sample datasets for more intuition. As mentioned in methodology chapter, at first, approximate clustering is applied on datasets (using APXDIST), and then clusters are revised based on *similarity in time* in the second step (using ED). Finally, the prototypes are merged in the third step based on *similarity in shape* using DTW.

In order to show the results visually, two datasets of the UCR repository (from TRAIN collection) are used in this experiment (the CBF and Coffee datasets). The results of clustering of the CBF and Coffee datasets are shown in the first step of MTC in Figure 6.1 and Figure 6.2. In this experiment, CBF dataset includes 30 instances. SAX6, APXDIST and k-Modes are used for clustering.

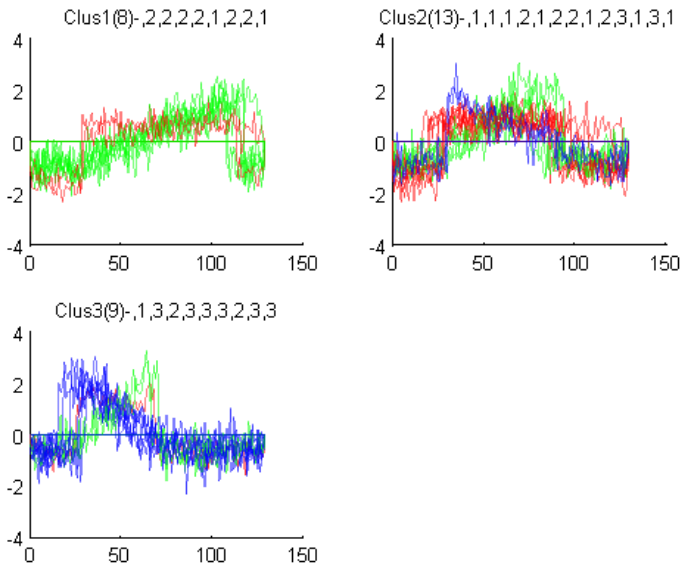


Figure 6.1: k-Modes clustering of time-series represented by SAX6 for CBF dataset (first step)

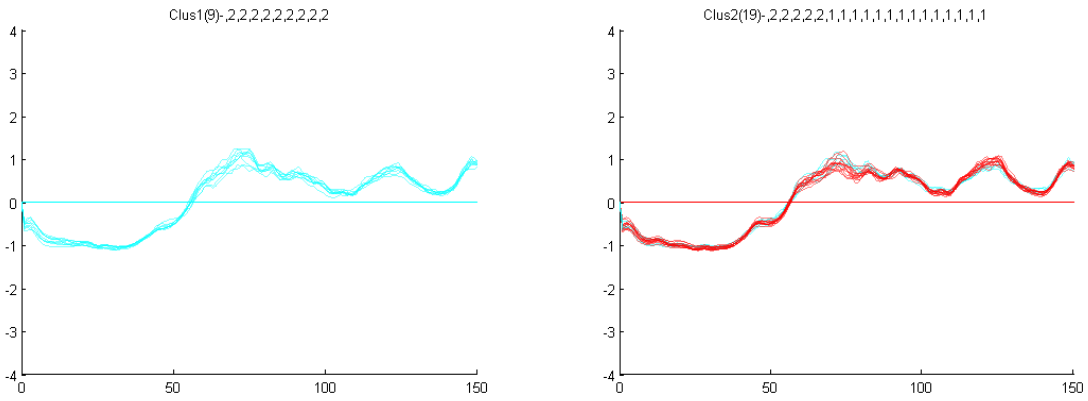


Figure 6.2: k-Modes clustering of time-series represented by SAX6 for Coffee dataset (first step)

Figure 6.1 and Figure 6.2 show that quality of clustering is low. For example, in CBF case, two time-series of second cluster has been wrongly combined with the first cluster. The main reason for low quality in this step is because of overlooking of data due to use of dimensionality reduction, and ignoring the shifts in the time-series utilizing a distance method which finds time-series which are similar in time.

Then, the second step of MTC is applied on sample datasets. The clustering solutions depicted in Figure 6.3 and Figure 6.4 corresponds to the second step of MTC clustering of CBF and Coffee dataset. For each dataset, the number of clusters shown in Figure 6.3 is more than the genuine clusters (ground truth). The additional clusters are the ones that contain outlier time-series or time-series related to other clusters and will be merged in the next step.

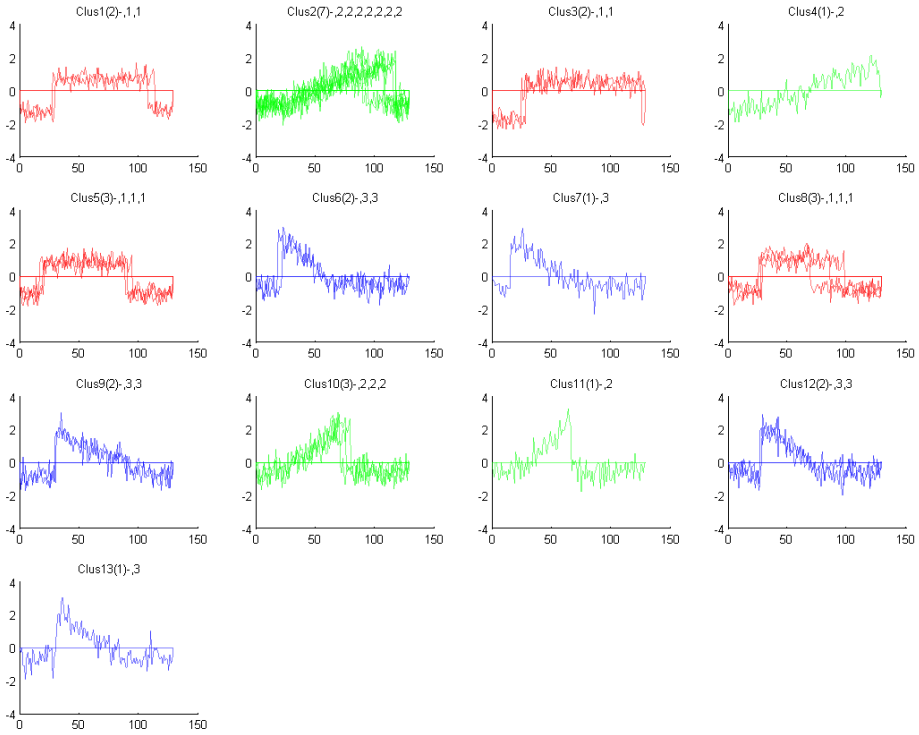


Figure 6.3: The clustering process in which MTC is able to find the genuine (pure) sub-clusters in the CBF dataset in the second step

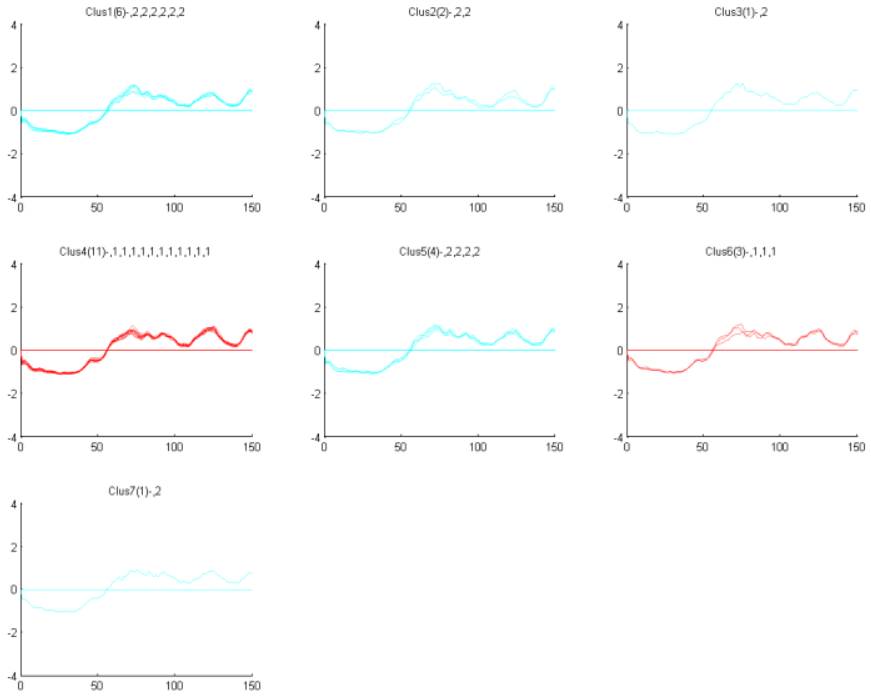


Figure 6.4: The clustering process where MTC is able to find the genuine sub-clusters in Coffee-train in the second step

Figure 6.3 and Figure 6.4 shows that MTC is able to identify the genuine clusters in both datasets. That is, MTC finds the most similar time-series in time in the second step. In the case of CBF (Figure 6.3), MTC finds 13 clusters, some of them correspond to the genuine clusters in the dataset, and some of them correspond to rare time-series (outliers). In the case of coffee-train (Figure 6.4), MTC finds seven clusters, each one corresponding to a genuine cluster in the dataset. For each cluster, a prototype is made which represents the whole time-series in the sub-clusters.

In the following figures, Figure 6.5 and Figure 6.6, the last result of CBF dataset containing 30 time-series and coffee dataset containing 28 time-series is shown, using the proposed model. In these examples, k-Medoids is utilized as merging scheme in the third step.

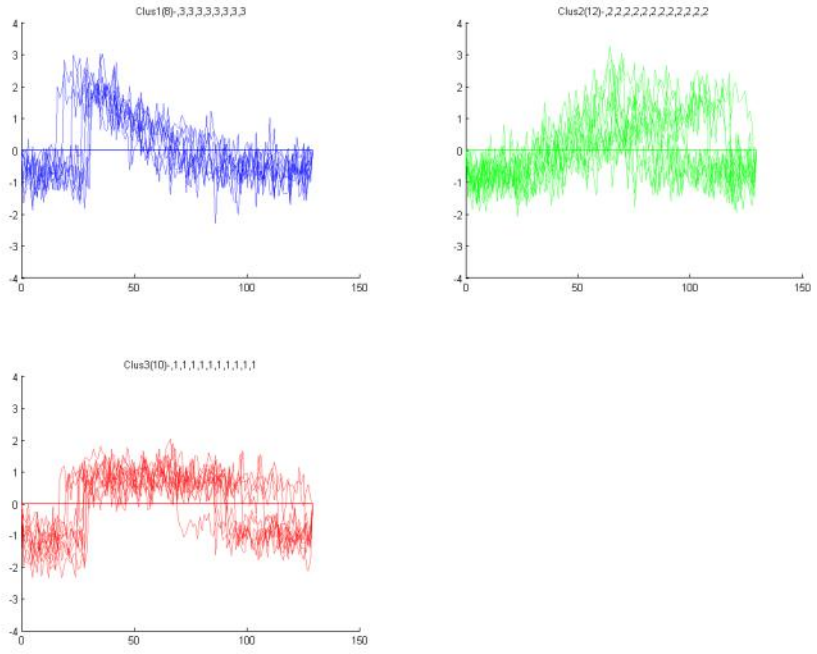


Figure 6.5: Clustering of time-series in CBF dataset using MTC

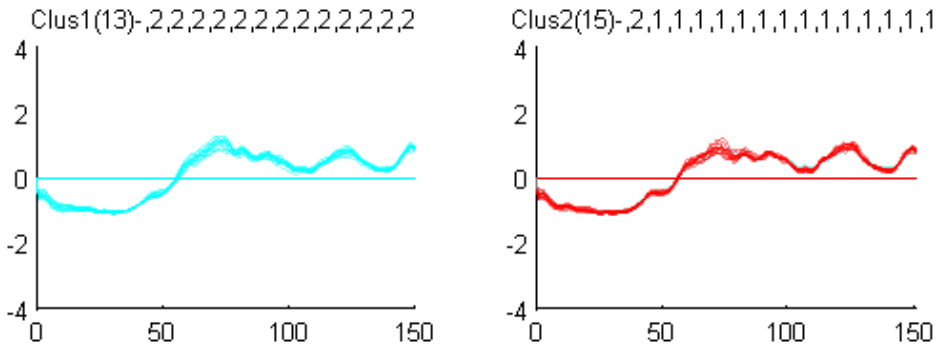


Figure 6.6: Clustering of time-series in Coffee dataset using MTC

As these experiments illustrate, MTC finds all the correct clusters. Similarly, MTC is very effective, and can make very high quality clusters for the Cofee dataset. Therefore, using these experiments, it was shown visually how MTC achieves very accurate clusters. In the next section, the accuracy of clustering is calculated objectively.

6.2.2 MTC Accuracy

In this experiment, accuracy of MTC is compared objectively with labelled time-series as ground truth. As mentioned before, nine clustering evaluation criteria are used for evaluating of time-series clustering algorithms, i.e., Jaccard, RI, ARI, F-measure, MNI, CSM, FM, Purity and Entropy. The details of each index were explained in the literature review in Section 2.7. Each of these clustering accuracy criteria has its own strengths and weaknesses in measuring quality, and there is no compromise of which measure is superior to other measure in the data mining community. Hence, to avoid biased evaluation, all of these measures (which all are between 0 to 1) are used and a conclusion is drawn based on the average of these measures. In this experiment 19 datasets from UCR repository are used to calculate the accuracy of MTC. For hierarchical merging of prototypes, “average” linkage and “single” linkage are used in this study as examples of hierarchical clustering group. The k-Medoids is used to show the results of partitioning clustering. k-Means is not used because of its weakness in finding the prototypes while similarity in shape is desirable (see Section 2.8.2 for more

details). Moreover, the introduced method for creating the arbitrary shaped clusters is used to merge the clusters with more than one prototype. Accuracy of MTC is shown in Figure 6.7. In this figure, the quality of MTC with different schemes for merging step is illustrated. In this experiment, the compression-ratio of SAX in the first step of MTC varies in each run from between 4 to 8. The random selection of initial clusters (in Ek-Modes) is addressed here by multiple runs of MTC to avoid obtaining the results by random chance (the algorithm was run 100 times), and the results are evaluated in front of ground truth.

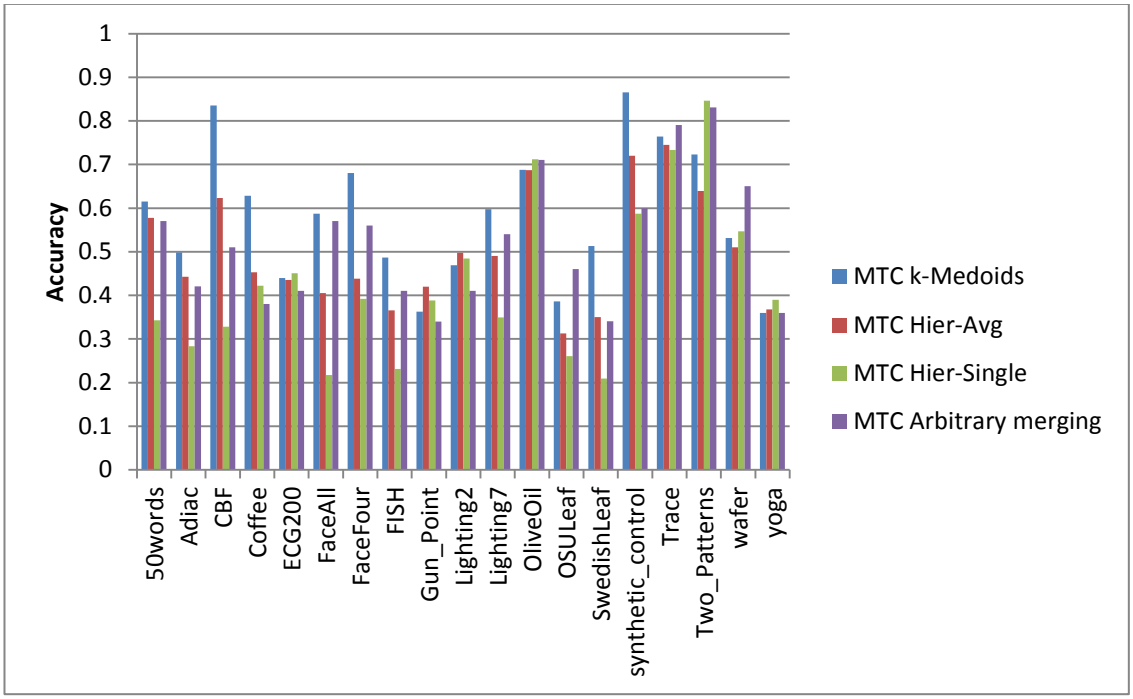


Figure 6.7: Clustering accuracy of MTC in front of ground truth using different schemes on various datasets

The results show that the accuracy of MTC is relatively high using all merging methods. However, in view of the classification results in UCR repository, it may seem that the results are not very significant or are not very accurate. For example, most of classification methods achieve up to 95% accuracy for all these datasets. Nevertheless, it should be considered that clustering is an unsupervised approach and classification is supervised. Therefore, clustering methods cannot often achieve the same accuracy of classification methods. Usually, clustering accuracy is lower (much lower) than

classification because it does not have training phase with train data and because of the random initialization of centroids. Therefore, to show the improvement of accuracy in clusters, the obtained results are compared with some other partitioning or hierarchical clustering of time-series data.

6.2.3 Comparing MTC with Partitioning Clustering

In this section, the results are compared with different methods of partitioning clustering. At first, k-Medoid is chosen which has been shown to be effective in finding clusters in time-series datasets (see Section 2.8.2). The k-Medoid is preferred to provide a fair situation for clustering of data in both conventional clustering and MTC.

In the beginning, raw time-series is taken into account as highest resolution. Although raw time-series cannot be used on the large time-series datasets, because of its high complexity problem (in real world problems), and as it usually contains outliers, the raw time-series is used ideally as input for comparison. To calculate distance between time-series data, ED and DTW are used as distance metric. In Figure 6.8, quality of MTC approach in front of quality of k-Medoids on raw time-series is shown. It is the result of 100 runs on the same datasets. Similar compression is also reported based on the purity measure (see Appendix E) to compare with some rival works.

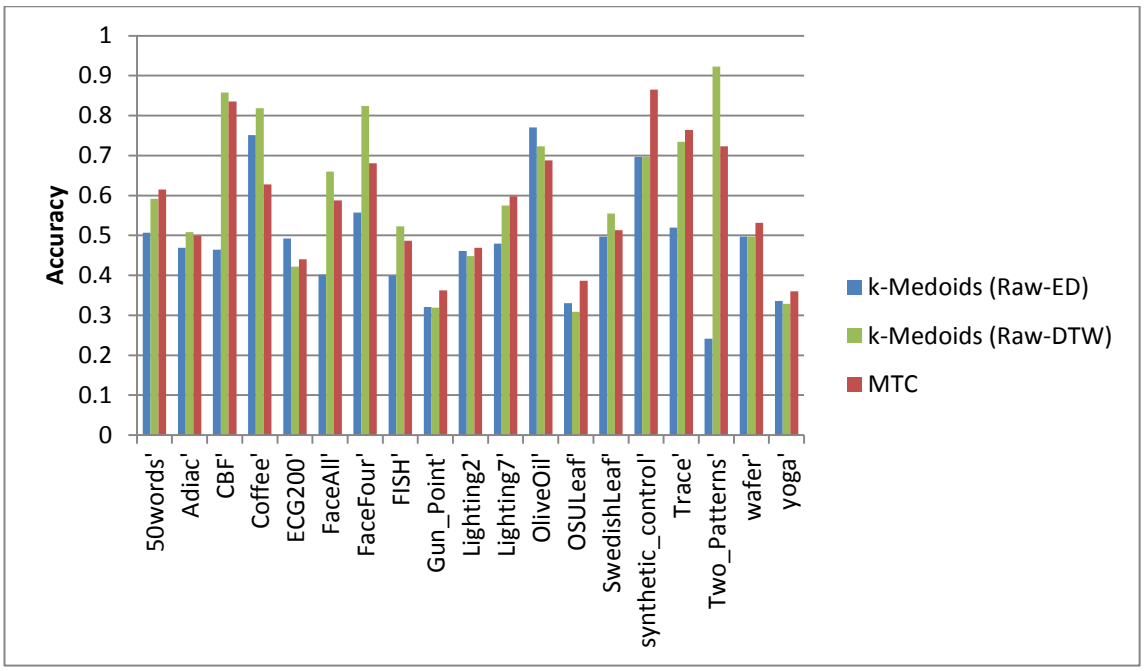


Figure 6.8: Quality of MTC approach in front of standard k-Medoids on raw time-series

First, using raw time-series and Euclidian distance, i.e., *k-Medoids(Raw-ED)*, MTC cannot find better clusters for Coffee, ECG, and Olive oil datasets, but it was able to find the right clusters for remaining 13 datasets. In this experiment, k-Medoids was applied on raw time-series. Raw time-series have quite good quality in this sense that they are not overlooked or approximated by representation methods. It is the point that is used to justify the cases whose quality of k-Medoids is better in comparison with MTC (e.g., Coffee, ECG, and Olive datasets). These datasets include time-series which are usually either very similar in time or their dissimilarities are mostly in amplitude (e.g., the samples of time-series of coffee and olive datasets depicted in Appendix J). Moreover, looking at dataset in the Table 5.1, it is understood that the size of these datasets (Coffee, ECG, and Olive) is relatively small (below 100 instances) which decrease the effect of using representatives in the MTC. That is, prototypes are representing only a few time-series, and the overlooking of data in making representations is more costly than increasing the reduction-rate (see Section 5.3.2.1) and handling shifts in time-series. However, in the most of the cases (in contrast with using DTW and MTC), *k-Medoids(Raw-ED)* has failed to find the accurate clusters

because of existing noise and shifts in time-series of other datasets which is very usual. That is, existing noise and shifts in the raw time-series directly influence similarity measure (ED), and so, leads to decrease the accuracy of k-Medoids. It would be even worse in k-Means because of the effect of the prototypes (centroids) on inaccuracy which is more obvious there. Therefore, MTC outperforms *k-Medoid(Raw-ED)*.

Second, as it is expected, using raw time-series and DTW, i.e., *k-Medoids(Raw-DTW)*, conventional k-Medoids works better in comparison with clustering using Euclidian distance, i.e., *k-Medoids(Raw-ED)*. It is because of the strength of DTW in handling shifts in time-series. However, MTC is still superior to *k-Medoids(Raw-DTW)* on some datasets. It is because of the mechanism of using a prototype of similar time-series instead of the original time-series which mitigates the effect of noises and outliers in time-series datasets while DTW is sensitive in front of outliers (because in DTW, all points have to be matched). Moreover, using raw time-series and DTW (i.e., *k-Medoids(Raw-DTW)*), is not a fair comparison because in the real world, it is not practically feasible due to its very high complexity. That is, just to calculate the confusion matrix (needed for clustering), it needs *N(N-1)/2* distance calculation, where *N* is the number of time-series. Additionally, considering the length of the time-series in the dataset as *d*, the number of running the instructions to calculate a distance measure for a pair of time-series is d^2. As a result, the complexity of only the distance matrix (not the whole clustering process) equals to $N(N-1)d^2/2$ which is very high. For example, in the Wafer dataset (from TRAIN set), given N=1000 and d=152, the number of executions of instructions is 11,540,448,000. However, the same process using MTC is around 177,084,440 (with SAX4 and reduction factor=90%, see Section 6.3 for more details). As a result, MTC is superior to *k-Medoids(Raw-DTW)*.

Based on literature review, most of the studies about clustering of time-series use a representation method to reduce the dimension of time-series before performing

clustering (see Section 2.8). Therefore, in the next experiment, as a fair comparison, the raw time-series are represented by SAX, and the MTC is compared in front of dimensionally reduced time-series. However, because SAX accuracy itself depends on the compression-ratio value, to provide also fair conditions, the raw time-series (for all datasets) are represented using three different compression-ratios ($\text{compression}_{\text{ratio}} = [4,6,8]$). Then, all datasets are clustered utilizing k-Medoids algorithm. As a result, for each dataset, mean of three accuracies is calculated as average accuracy of k-Medoids. Likewise, as MTC also can accept different resolutions in the first step, the same resolutions have been used for MTC. The mean of accuracies is considered as the quality of clustering of data using MTC. The following chart shows the comparison of average quality of k-Medoids with the average accuracy of MTC which are compared against the ground truth (see Figure 6.9).

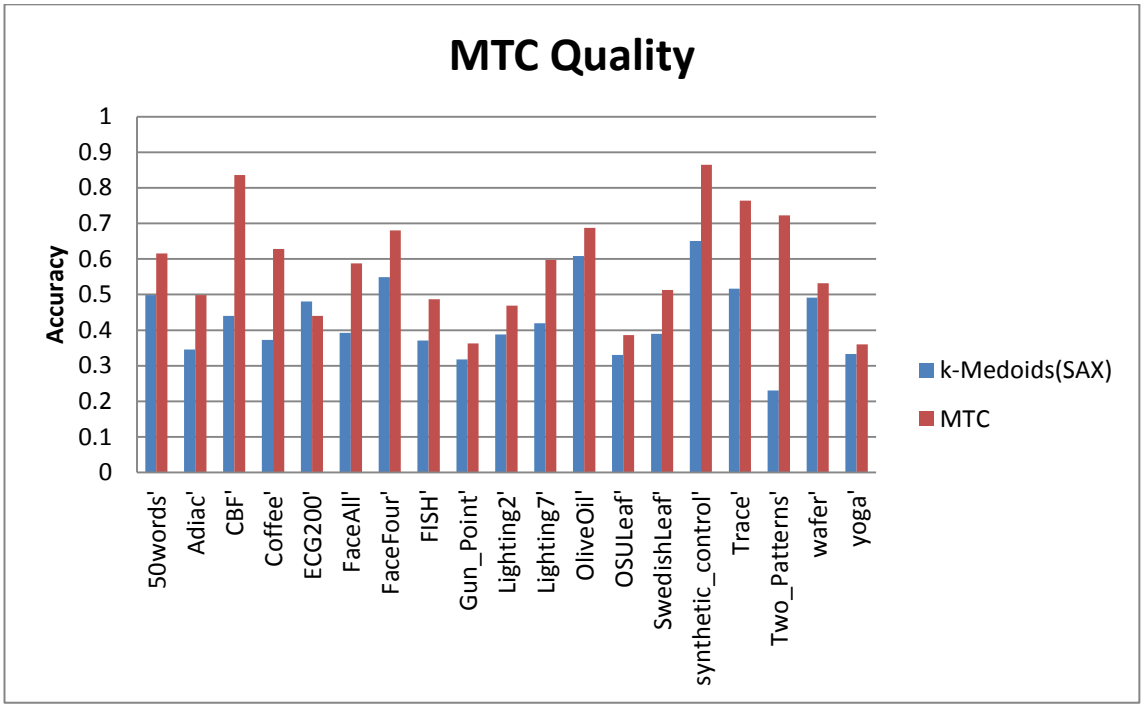


Figure 6.9: Quality of MTC (k-Medoids) in front of ground truth for UCR dataset

As Figure 6.9 shows, as expected, the accuracy of MTC is much better than k-Medoids for all datasets. Moreover, the maximum quality has been calculated for k-Medoids and the result is reported in Appendix G. It is the proof of the researcher's claim that the

MTC model can outperform the conventional algorithms which use dimensionality reduction approaches for clustering purpose.

6.2.4 Comparing MTC with Hierarchical Clustering

In this section, accuracy of MTC is compared with hierarchical clustering algorithms. Hence, hierarchical scheme is utilized in the third step of MTC. In the case that a hierarchical clustering algorithm is used as merging scheme for the third stage of MTC, a dendrogram is produced of possible clustering solutions at different steps of granularity. Because merging is on prototypes of each sub-cluster which are small (in comparison with original data), it is very easy to visually show the hierarchy of clusters which is very advantageous. For each dataset, two different clustering solutions are experimented, i.e., Hierarchical clustering with *average* linkage and *single* linkage.

At first, the conventional hierarchical clustering is experimented on the CBF dataset (from TRAIN set). In this experiment time-series are transformed to SAX4 representation. Figure 6.10 shows the generated dendrogram using hierarchical clustering (average linkage) of CBF dataset.

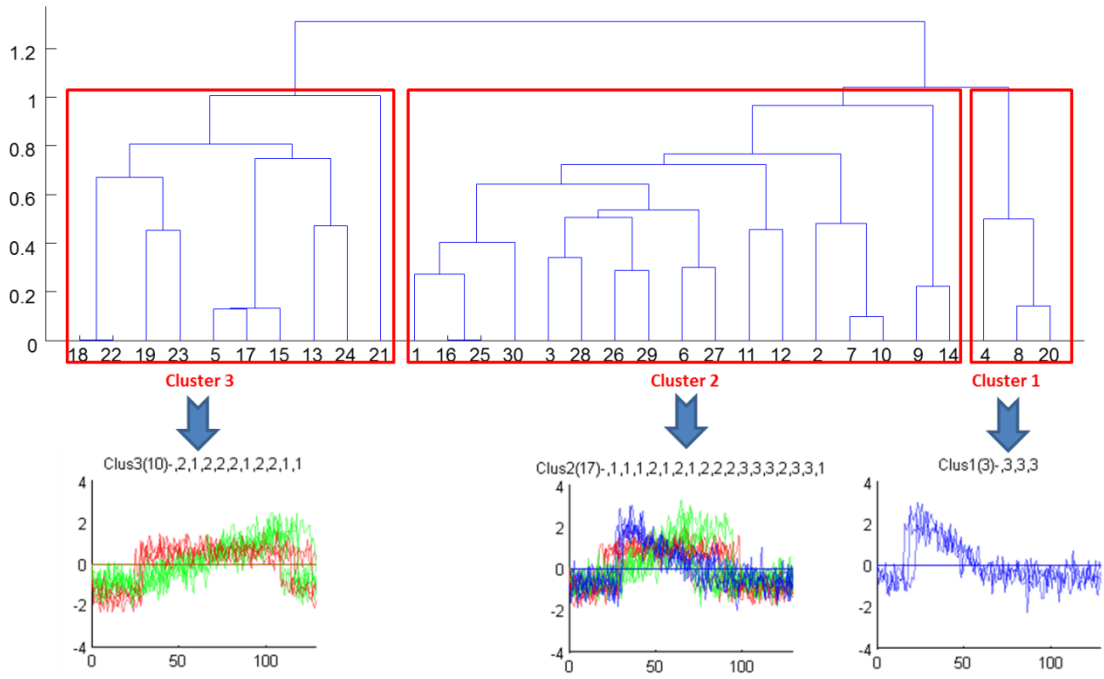


Figure 6.10: Dendogram of hierarchical clustering of CBF using average linkage

As it is seen, in this clustering solution (Figure 6.10), hierarchical clustering merges clusters wrongly. Hierarchical clustering has failed to select the correct pair of clusters to merge them together, and as a result, the red-colored time-series and green-colored time-series are merged in the first and second clusters. Table 6.1depicts the quality of final clustering result.

Table 6.1: The quality of wrong clustering of CBF using Hierarchical clustering by average linkage and SAX4

RI	ARI	Purity	CEntropy	f-measure	Jacard	FM	NMI	CSM	Hier(Linkage.Avg) (SAX4)
0.55	0.09	0.53	0.2	0.71	0.28	0.45	0.23	0.46	0.39

In contrast, a successful experiment made from MTC methodology is shown in the following. At first the result of MTC is depicted in the first step where k-Modes and SAX4 should be used to generate pre-clusters. However, to provide exactly the same condition, the pre-clusters generated in the conventional hierarchical clustering are reused here as depicted in Figure 6.11. Actually, this clustering solution corresponds to the earliest dendrogram where the hierarchical process failed to merge sub-clusters which belong to two different classes (better pre-clusters are made using k-Modes as experiments also shows).

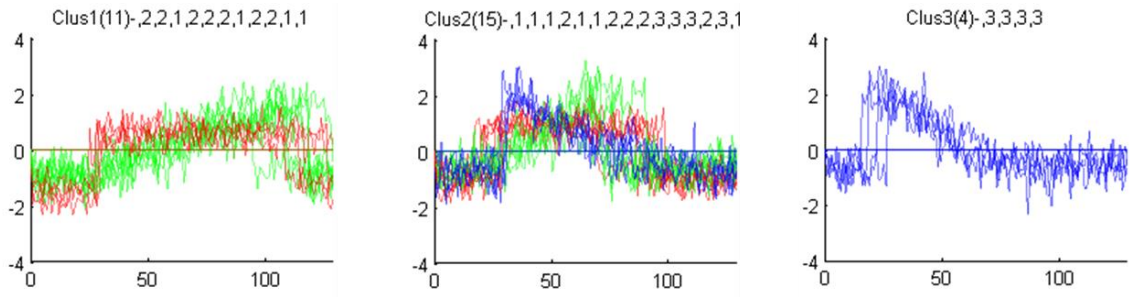


Figure 6.11: Pre-clustering of the time-series related to CBF made by hierarchical clustering

Then the pre-clusters are refined in the second step. Figure 6.12 shows the process of making sub-clusters from pre-clusters.

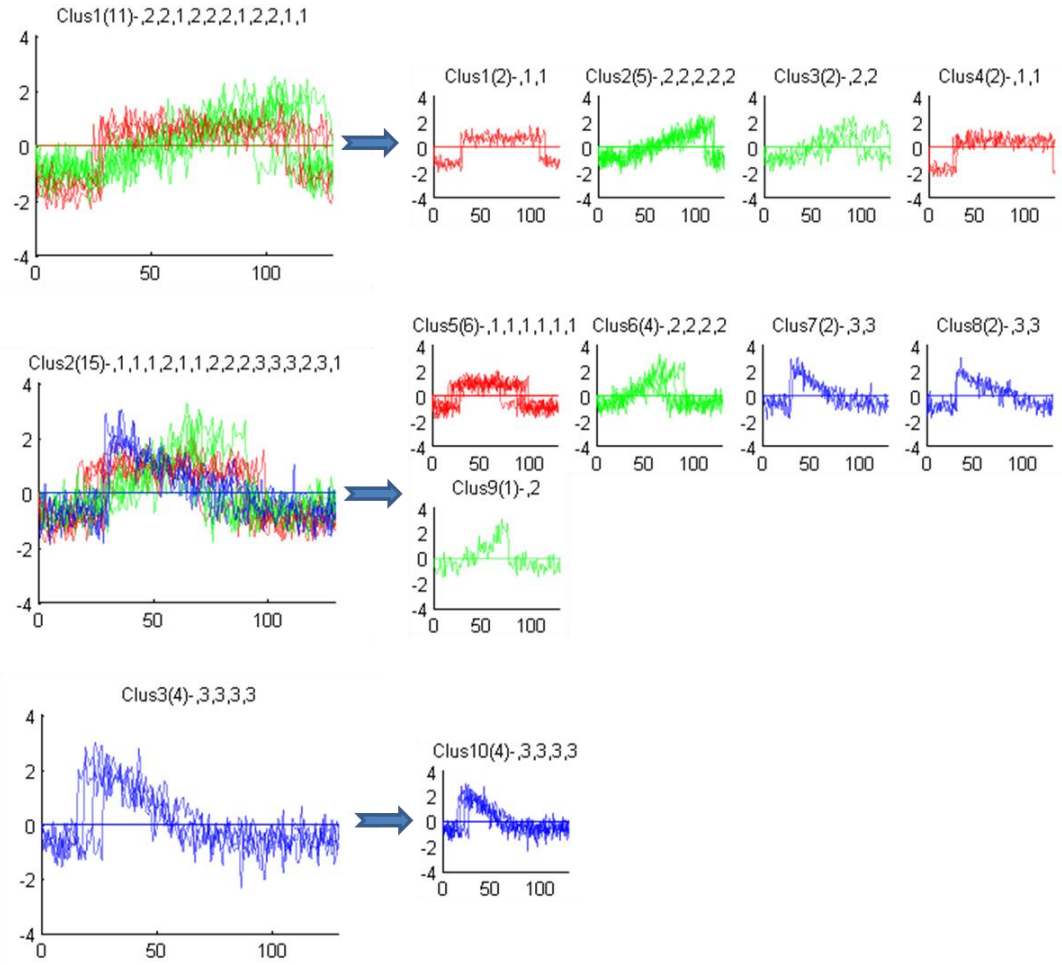


Figure 6.12: Revising pre-clusters: generating the sub-clusters

As Figure 6.12 represents, pre-clusters are broken into the sub-clusters of similar time-series in time. It is the result of applying PCS on pre-clusters where it purifies the clusters (see Section 4.4.2 for more details). As it is shown, all sub-clusters in this example are quite pure. Consequently, for each sub-cluster a prototype is made which represents the whole sub-cluster and this is depicted in Figure 6.13.

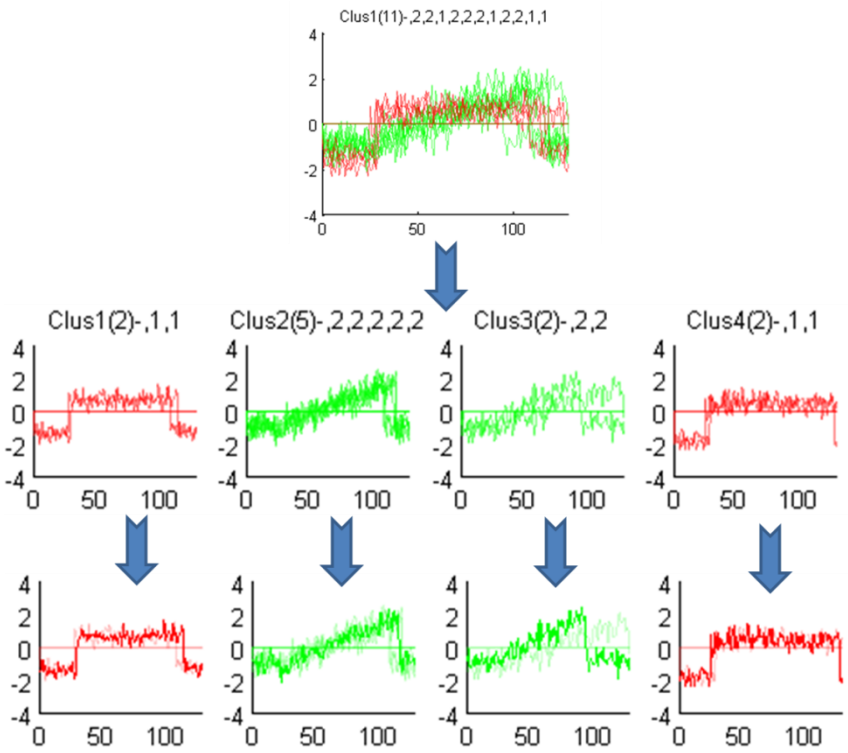


Figure 6.13: Selecting a time-series as the prototype of a sub-cluster

At last, using the hierarchical scheme, the prototypes are clustered in the third step. The result is clusters of prototypes which are made based on the similarity in shape.

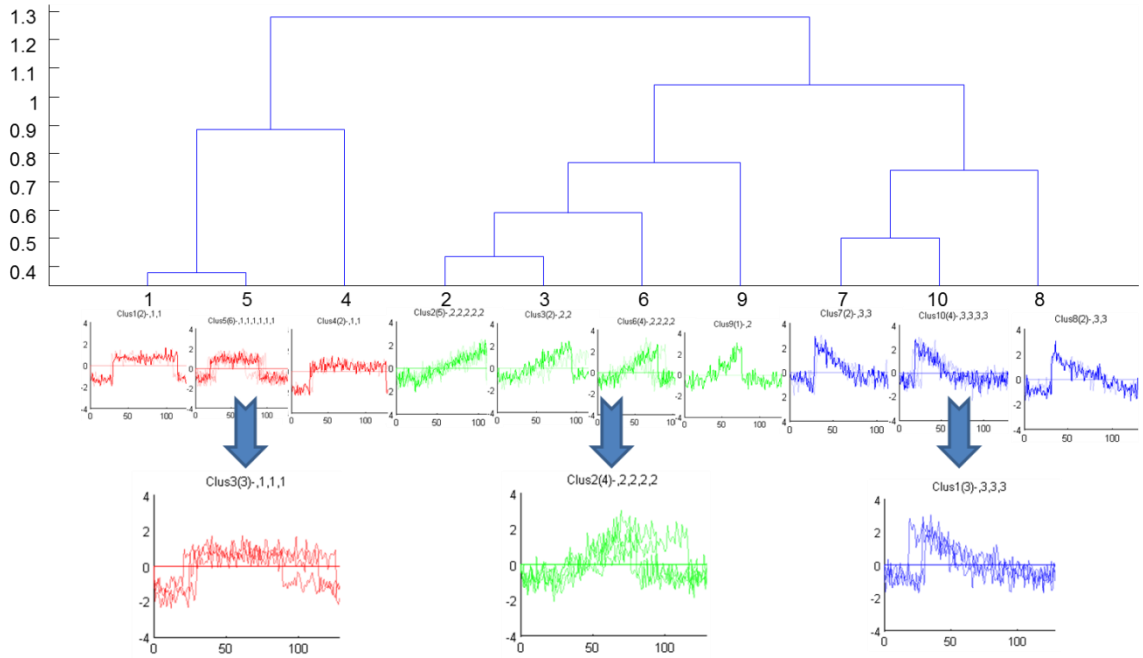


Figure 6.14: Third step of MTC: Merging prototypes

As Figure 6.14 shows, hierarchical clustering is performed only on the provided prototypes which are less than the original data. As a result, calculating of similarity in

shape (which is very costly) is carried out only on a few sets of time-series. Then the prototypes are replaced through mapping activity as it is illustrated in see Figure 6.15.

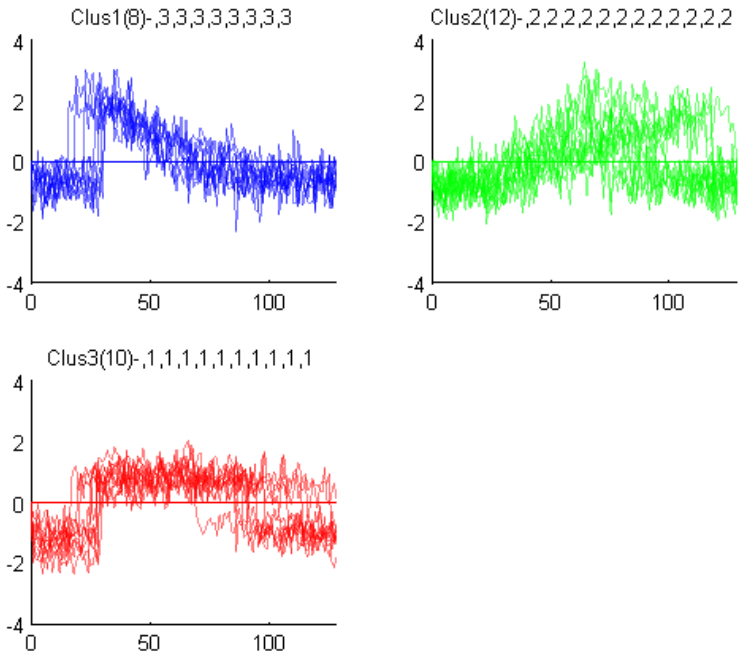


Figure 6.15: The final result of MTC on CBF dataset

As Figure 6.15 shows, using this approach, a high accuracy (even up to 100%) can be obtained in some datasets. To show how effective is this approach, the proposed model is applied on different datasets. The following diagram (Figure 6.16) shows the result of MTC clustering by the hierarchical scheme with average linkage.

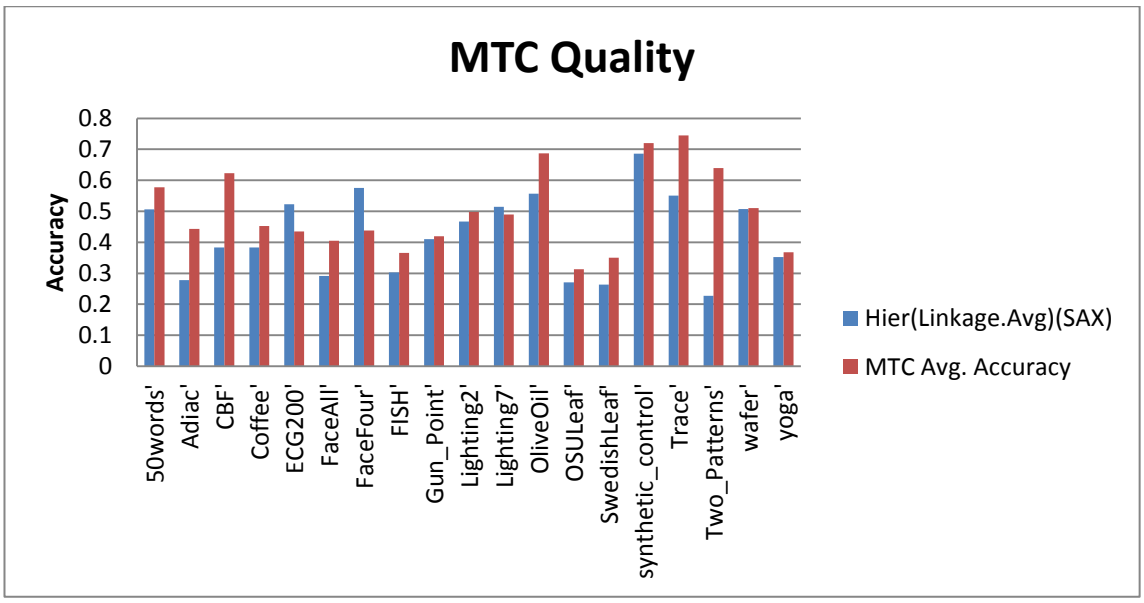


Figure 6.16: Average quality of MTC approach in front of running hierarchical clustering (average linkage) on dimensionally reduced time-series.

As the results in Figure 6.16 shows, the accuracy of MTC for most of the datasets outperforms the conventional hierarchical algorithm using the average linkage. Hierarchical clustering using single linkage also is a well-known clustering algorithm that is good at handling non-elliptical shapes but however, it is sensitive to noise and outliers (Tan et al., 2006). Figure 6.17 shows the result of running MTC where hierarchical clustering with single linkage is used.

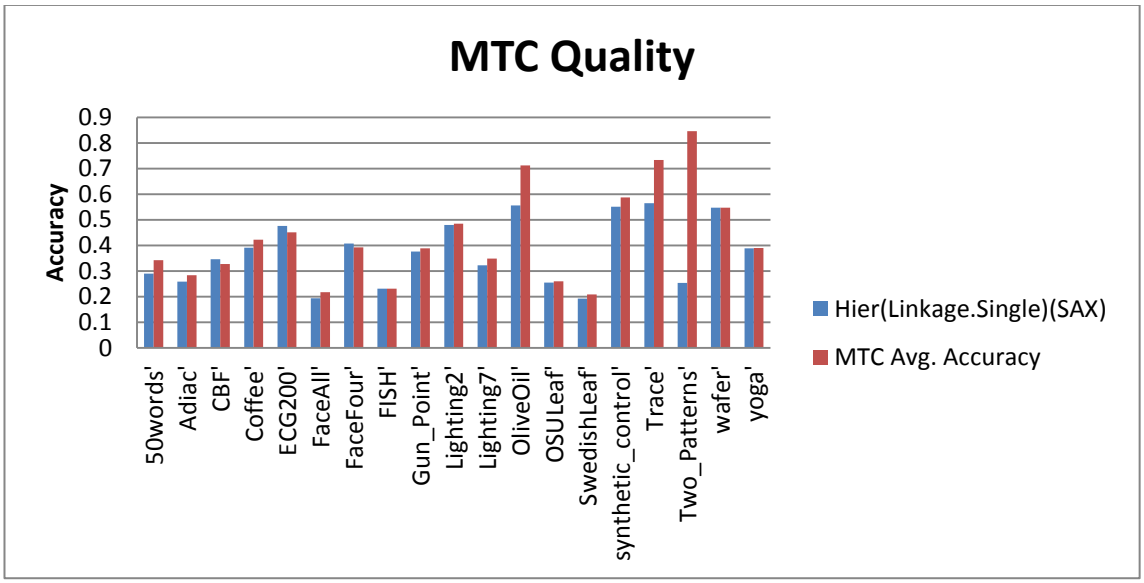


Figure 6.17: Average quality of MTC approach in front of running hierarchical clustering (single linkage) on dimensionally reduced time-series.

As expected, in this approach also MTC is outperforming the single linkage for most of the cases. To sum up, it is understood that using MTC and hierarchical schemes, the prototypes of time-series can be clustered which generate a dendrogram which is very useful for data structure understanding. Moreover, as it was shown for partitioning scheme, in hierarchical scheme also, approximated sub-clusters, has a less destructive effect on the accuracy of the final clusters compared with the use of approximate time-series, and generates more accurate clusters than them (i.e., approximate time-series). Therefore, user does not need to reduce the dimension of time series to a high extend, and as a result, the data is not overlooked in the process of clustering.

6.2.5 Comparing MTC with Multi-step Models (Rival Models)

One of the novel works which is near to the proposed model in this study is a two-level approach proposed by Lai et al. (2010), so called 2LTSC, which was discussed in the literature review (Section 2.8.6). In this work, SAX transformation is used as a dimension reduction method and CAST as clustering algorithm in the first level. In the second level, to calculate distances between time-series, Dynamic Time Warping (DTW) has been used for varying length data, and Euclidean distance for equal length data. To compare 2LTSC with MTC, the fair conditions are taken. First, both models are applied on the similar datasets from the UCR repository as benchmark dataset. Second, the 2LTSC works with CAST algorithm where the number of clusters is determined indirectly by a threshold. Hence, after running the 2LTSC, the similar number of clusters (generated by 2LTSC) is used also for MTC. Third, similar parameters for SAX representation are used to generate the results. Four, because 2LTSC is very sensitive to threshold parameter of CAST algorithm, the best result of 2LTSC is captured as the quality of the model. Figure 6.18 shows the result of clustering for both approaches.

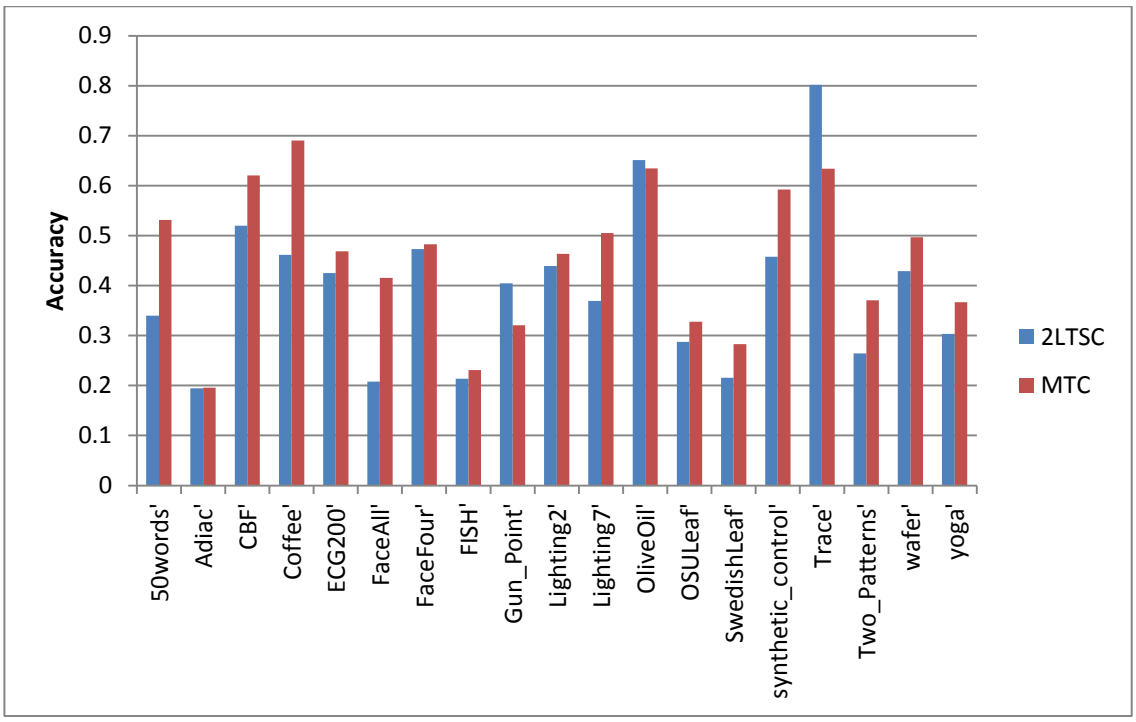


Figure 6.18: Comparison of 2LTSC and MTC in front of ground truth for test dataset

In addition to the high complexity of CAST and its sensitivity in front of its parameter as explained in the literature review (Section 2.8.6), as it is presented in Figure 6.18, the quality of MTC is superior to 2LTSC for most of the datasets. The main reason for this priority is using more accurate approach in the first step of MTC in comparison with 2LTC. As mentioned, in MTC model, SAX is used with APXDIST which is superior to MINDIST used in 2LTSC. As a result, the quality of MTC is increased after revising the pre-clusters in the second level. Moreover, in the third level of MTC, the sub-clusters are merged again which makes the generated cluster structure more similar to the ground truth.

Comparing the accuracy of MTC in this experiment (Figure 6.18) with the results of MTC using k-Medoid scheme (Figure 6.9), the accuracy of MTC varies a bit. It is because of different cluster numbers in this experiment. That is, as mentioned, the number of clusters in this experiment is based on the results of 2LTSC.

Another study which is performed in more than one step is a work presented by (X. Zhang et al. (2011) and was discussed in the literature review (Section 2.8.6). As

mentioned, they proposed a new multi-level approach for shape based time-series clustering where some candidate time-series are chosen and clustered. Candidates are chosen from a one-nearest neighbour network by the triangle distance between time-series data. To choose the candidate, they analysed different orders of neighbour network (graph). Then, hierarchical clustering is performed on selected candidate time-series using DTW. To compare this approach (so called graph-based) with the MTC model, the maximum quality of MTC clustering in front of different orders of the nearest neighbour network is calculated and shown in Figure 6.19. To provide fair conditions, the order of 2 to 5 is considered for graph-based approach which provides a reasonable reduction in the second layer (see Appendix H).

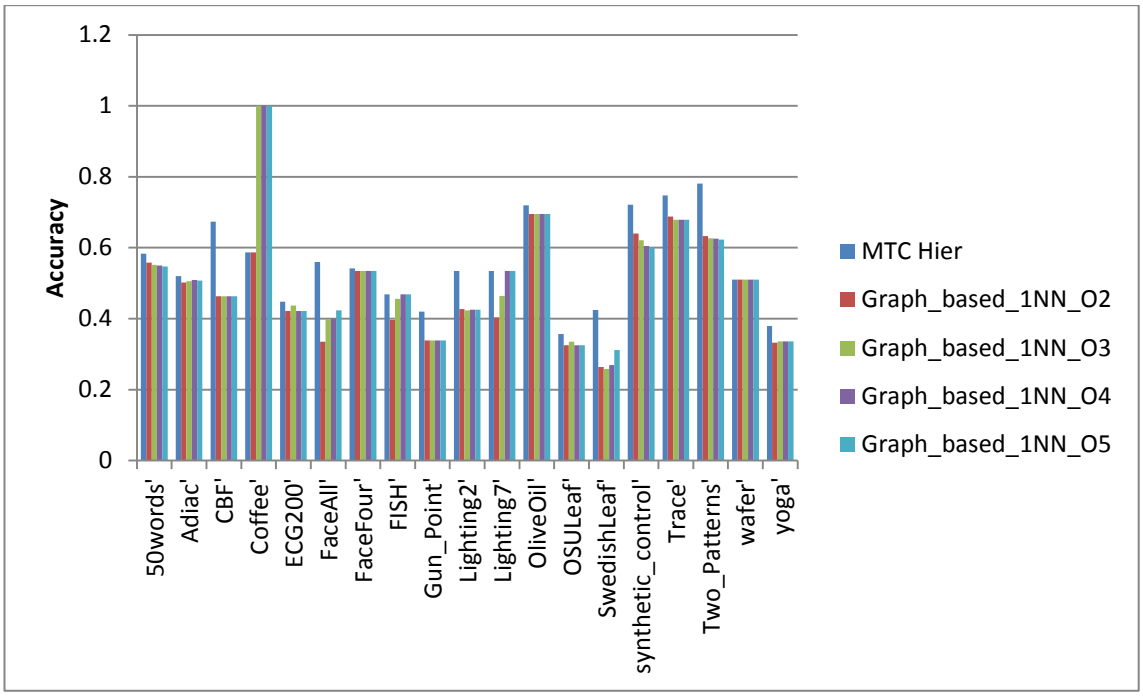


Figure 6.19: Quality of clustering using graph-based approach in front of MTC

As the result shows, in the most of the cases (15 cases), MTC is superior to graph-based algorithm. Additionally, it should be noted that in graph-based approach, the first level clustering is performed on raw time-series (which is potential to generate noisy clusters) and needs to make a nearest neighbour graph which is very costly. However, it can be advantageous for datasets where the similarity in time is essentially very important such

as Coffee dataset (see Figure 6.16). To summarize, the proposed model (MTC), can beat the rival approaches, even with lower time complexity.

6.2.6 Evaluation of MTC on Large Datasets

In order to further confirm the effectiveness of MTC, some experiments are carried out on large synthetic datasets. For this purpose, CC dataset and CBF dataset with different cardinalities are generated up to 12000 records, and accuracy of clustering is computed using MTC for each dataset.

In order to evaluate the results of the proposed model on large datasets, MTC (with different schemes of merging) was experimented in front of conventional k-Medoids and hierarchical approach (with single and average linkages). To achieve the best results, highest resolution of time-series, i.e., raw time-series, with the Euclidian and DTW distances are used for clustering by k-Medoids and hierarchical approach. For fairness, the average quality of running MTC with two constraint values of DTW (i.e., $DTW_r = 50\%$ and $DTW_r = 100\%$) are considered as the quality of MTC for all cardinalities of the dataset. The average quality of MTC with different schemes is depicted in Figure 6.20 and Figure 6.21.

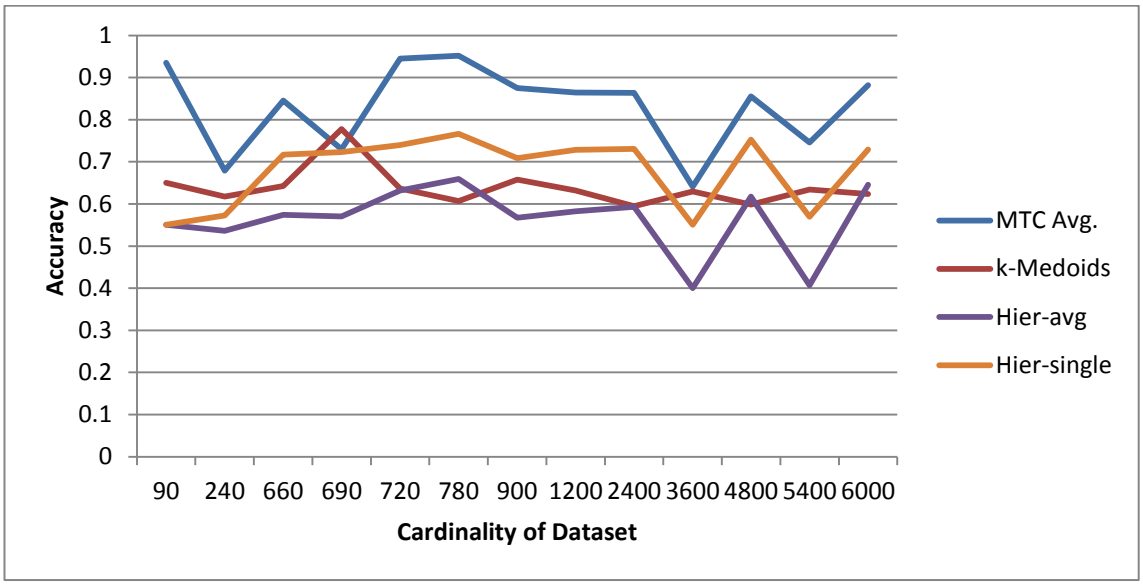


Figure 6.20: Accuracy of MTC in front of other algorithms for CBF dataset

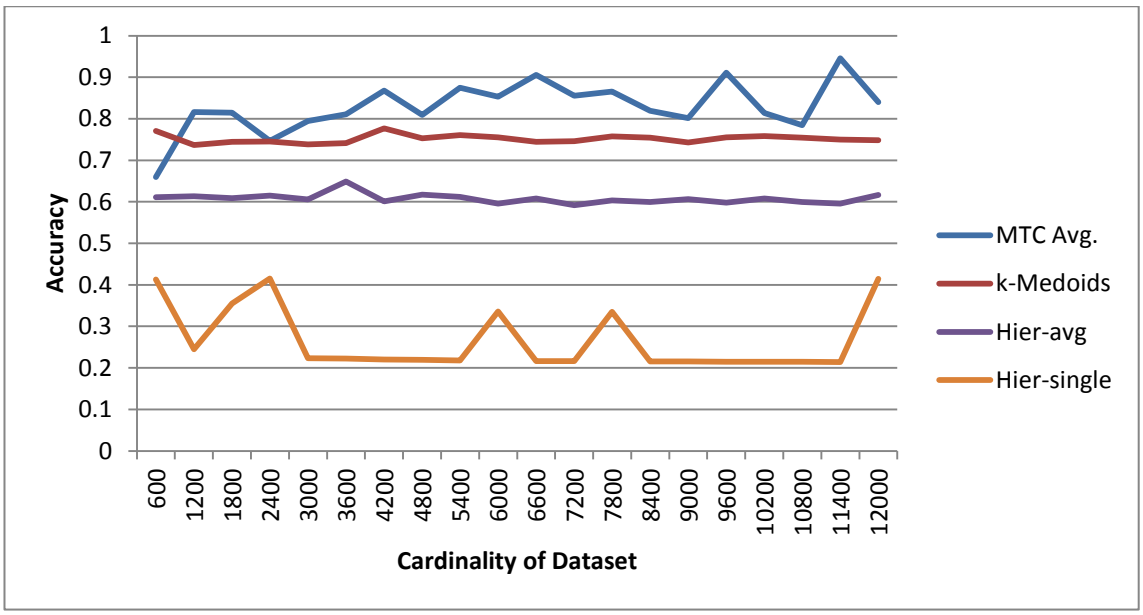


Figure 6.21: Accuracy of MTC in front of other algorithms for CC

As the result shows, the quality of MTC is superior to other algorithms. However, the fluctuation of MTC in both datasets is a bit more than other algorithms (except hierarchical algorithm using single linkage). It is the effect of, first, showing the result by averaging of accuracy per different parameter values of SAX, and second, using prototypes of data instead of original time-series, i.e., the error of the second step in purifying the clusters. However, for most of the cardinalities of the dataset, it is seen that the minimum quality of MTC is still more than other algorithms. Therefore, it can be concluded that in most of the datasets, MTC can generate better results. Therefore, for clustering large time-series datasets, it was understood that there is no need to use very low dimensional time-series; instead, the clustering can be applied on smaller sets of high dimension time-series by prototyping. That is, the cost of using representatives is much less than dimension reduction in terms of accuracy.

6.2.7 Evaluation of MTC Using Internal Criteria

In this section, accuracy of MTC is evaluated using internal criteria which are more realistic in comparison with external criteria, because class labels are not available in the most of the real world problems. Two datasets from financial systems are chosen to

evaluate the proposed model while ground truth does not exist for them. Financial domain is a proper system for clustering because there are many financial tasks in this domain which need pre-processing or pattern recognition, for example, forecasting stock market, currency exchange rate, bank bankruptcies, understanding and managing financial risk, trading futures, credit rating, loan management, bank customer profiling, and money laundering analyses as core financial tasks for data mining (Nakhaeizadeh, Steurer, & Bartlmae, 2002). Corresponding to these tasks, clustering and outlier detection are appropriate approaches. For instance, clustering of time-series related to transactions of customers can answer the following questions:

1. Can one determine a customer category given a historical record of the transaction history?

2. How are the movements of account balance across the various customers?

As an example, a dataset related to a Bank in Malaysia called Banking Transaction Dataset (BTD) is used for this experiment. Challenging feature of this dataset is that time-series are very close to each other and it seems that the clusters have different densities. BTD consists of 6802 time-series, and each series of BTD is composed of 360 time points, each of which denotes one customer’s daily transaction amount (balance or counts) in one year.

As explained, compression-ratio can vary from 4 to 8. The compression-ratio=6 is used here to symbolize the time-series and to be fair in terms of the resolution. Then, MTC is applied to cluster the dataset. Not like CBF and CC datasets, the series of this dataset are not clustered in advance, so class labels cannot be used to verify the final clustering results. Moreover, considering the unsupervised nature of clustering and unavailability of number of clusters, the number of clusters should be determined by the user. Essentially, the right number of clusters depends on the distribution's shape and scale in

the dataset, as well as the clustering resolution required by the user (Han & Kamber, 2011). For this dataset, a simple way is used to determine the number of clusters. It is set to about $\sqrt{N/2}$, where n is the number of time-series in the dataset. That is, in each cluster there are around $\sqrt{2N}$ time-series.

In Figure 6.22, some of the generated clusters and their prototypes are illustrated.

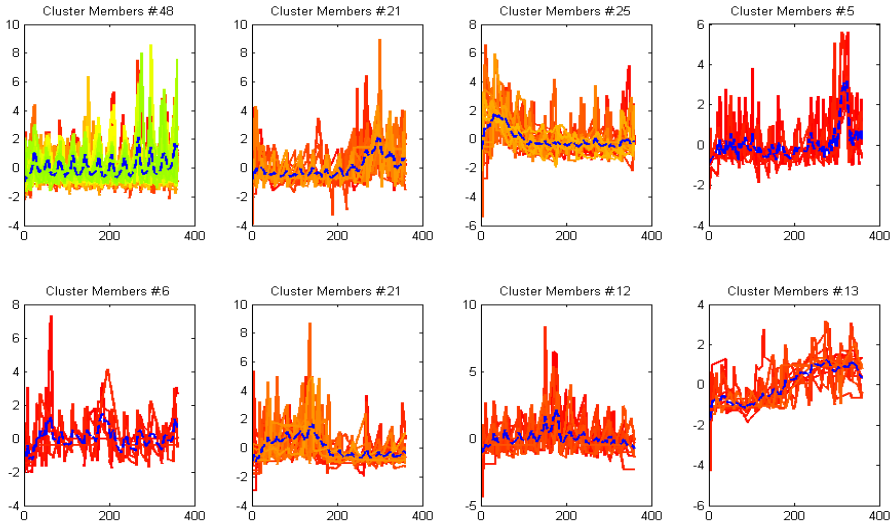


Figure 6.22: Sample clusters of time-series related to transaction of customers

The results show different groups of bank customers which are interpretable for banking system experts. For example, the first cluster is related to the customers who deposit their salary in their account (or it is transferred to their account by an employer) in the first day of each month. However, it is necessary to use some validity criteria to show the superiority of MTC. In order to prove that the proposed model is more efficient than utilized conventional algorithms for clustering whole time-series, conventional k-Medoids is adopted. Different amounts of records (cardinalities) from the dataset are collected to show how MTC works in terms of accuracy. As mentioned, clustering is an unsupervised learning technique and there are no predefined classes to compare the clustering results of different clustering algorithms. For evaluation of such clusters in terms of accuracy, the most common measure for internal criteria, Sum of Squared

Error (SSE), is used (Han & Kamber, 2011). For each time-series, the error is the distance to the prototype of nearest cluster. The following formula is used to calculate SSE:

$$\mathrm{SSE} = \sum_{j=1}^{K} \sum_{F_i \in C_j} (\mathrm{dist}(F_i, R_j))^2 \qquad 6.1$$

where, F_i is a time-series in cluster C_j and R_j is the prototype of cluster C_j. Here, dist() is the DTW distance which had been shown that is very useful for making natural clusters. The result illustrated in Figure 6.23 is related to the average of SSE of 10 times run of the MTC and k-Medoid with different resolutions of SAX (different compression-ratios).

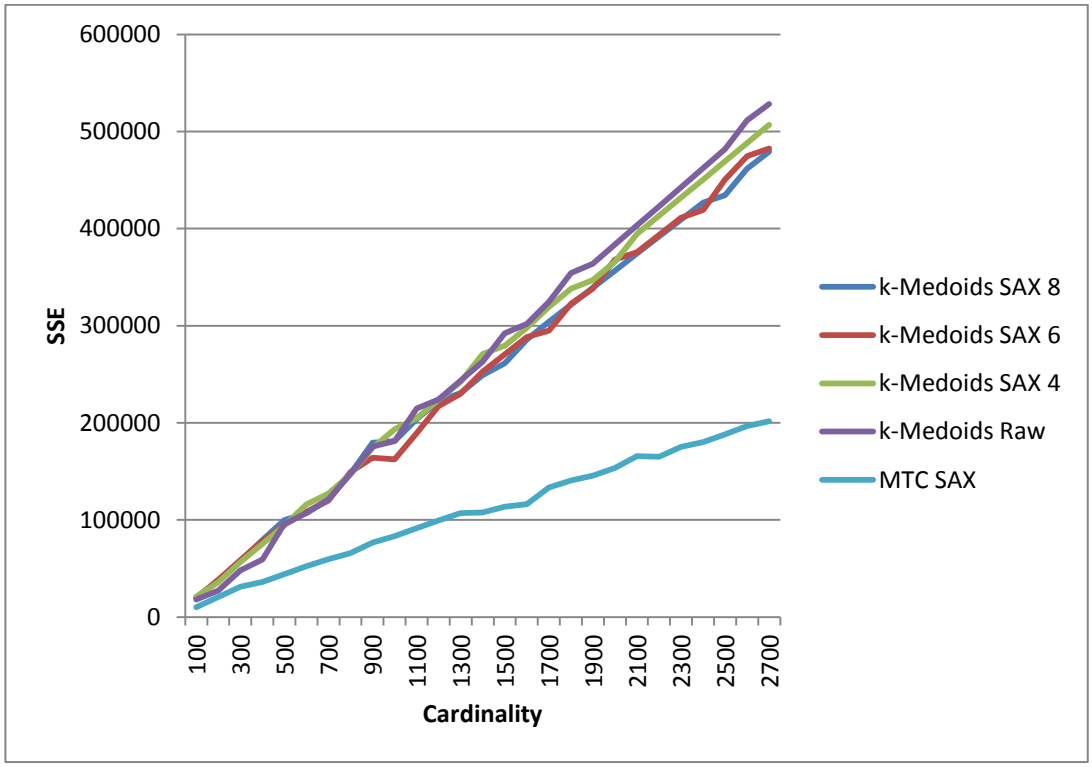


Figure 6.23: Accuracy of MTC in front of different cardinalities of BTD

The result in Figure 6.23 reveals that SSE of MTC is less than conventional clustering using SAX representation. As the results represent, the quality of MTC is high even in comparison to raw time-series (not dimensionality reduced data). The superiority of MTC is due to use of the similarity in shape while conventional clustering algorithms

are using the similarity in time. Although, there is no significant difference in the quality of MTC and k-Medoids (Raw) in the small cardinalities of the dataset, the result clearly shows that as the cardinality is increased, the difference in SSE also is increased and this means MTC works well with high cardinality datasets. In fact, the effect of using DTW shows its impact in higher cardinality, while the effect of using raw time-series is decreased as the size of the dataset (cardinality) grows up.

Furthermore, the experiment is repeated for another dataset with different characteristics. KLSE (Kuala Lumpur Stock Exchange) is End Of Day (EOD) data related to the stock exchange of Kuala Lumpur, Malaysia. The historical data is provided by www.klseeod.com which is retrieved from public websites and is published freely for educational purposes.

The objective of this task is to cluster the stock exchange time-series in order to find the clusters of companies which are similar in shape of their stock price. The stock exchange of companies is from different markets such as Main Market, ACE Market, Structure warrants, ETF, and Bound. These companies belong to different categories, such as Property, Plantation, Mining, etc. Finding groups of similar companies can be used for prediction tasks. For this experiment, a set of time-series related to one year (2010) is chosen. The number of companies (time-series) in this experiment is 870 and each time-series consists of 240 points.

At first, the time-series are normalized using Z-normalized to provide fair conditions. Then, the number of clusters determined by $\sqrt{n/2}$ rule because it is not pre-defined in advance. Finally, MTC is applied on whole time-series data. Figure 6.24 shows a sample of the constructed clusters related to the stock exchange time-series.

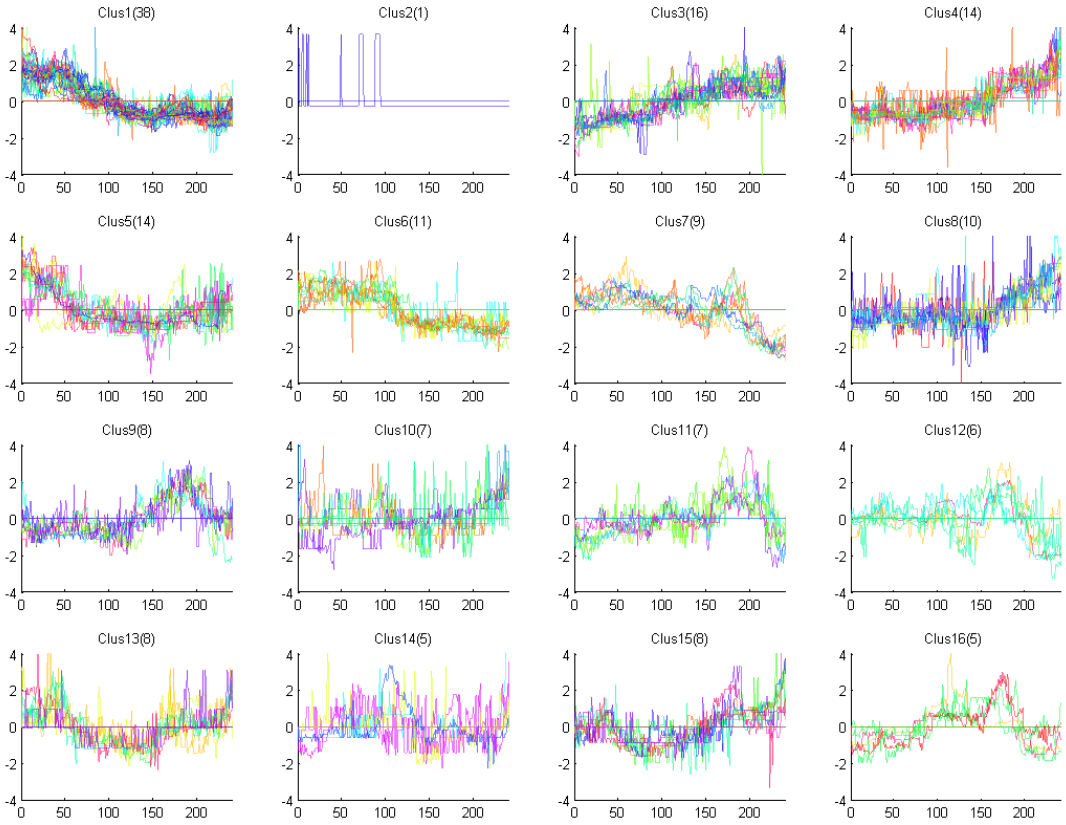


Figure 6.24: A sample of clustering of time-series related to KLSE

Different clusters that are generated by MTC are depicted in Figure 6.24. The clustering results show that, first, MTC can reveal the outliers in data. For example second cluster includes only one time-series which can be considered as outlier company or missing data. Second, as expected, MTC generates the clusters of different companies with the similar stock exchange. However, as the figure shows, the generated clusters are not very clear and useful because usually there are some shifts in the exchange stocks. Nevertheless, as explained, MTC has the ability of generating dendrogram of time-series using the prototypes of the second step. It gives the experts the insights in different domains. That is, using the dendrogram of prototypes, different patterns of stock exchange is revealed which also can be utilized in classification tasks.

Additionally, in lower levels of the dendrogram, similar groups of time-series can be used as a tool for prediction. This capability comes from the strength of MTC in finding similar time-series in shape. Figure 6.25 shows a sample of clustering of companies

based on the similarity of their stock exchange in shape. This kind of dendrograms can be used to predict stock exchange based on similar companies in a cluster.

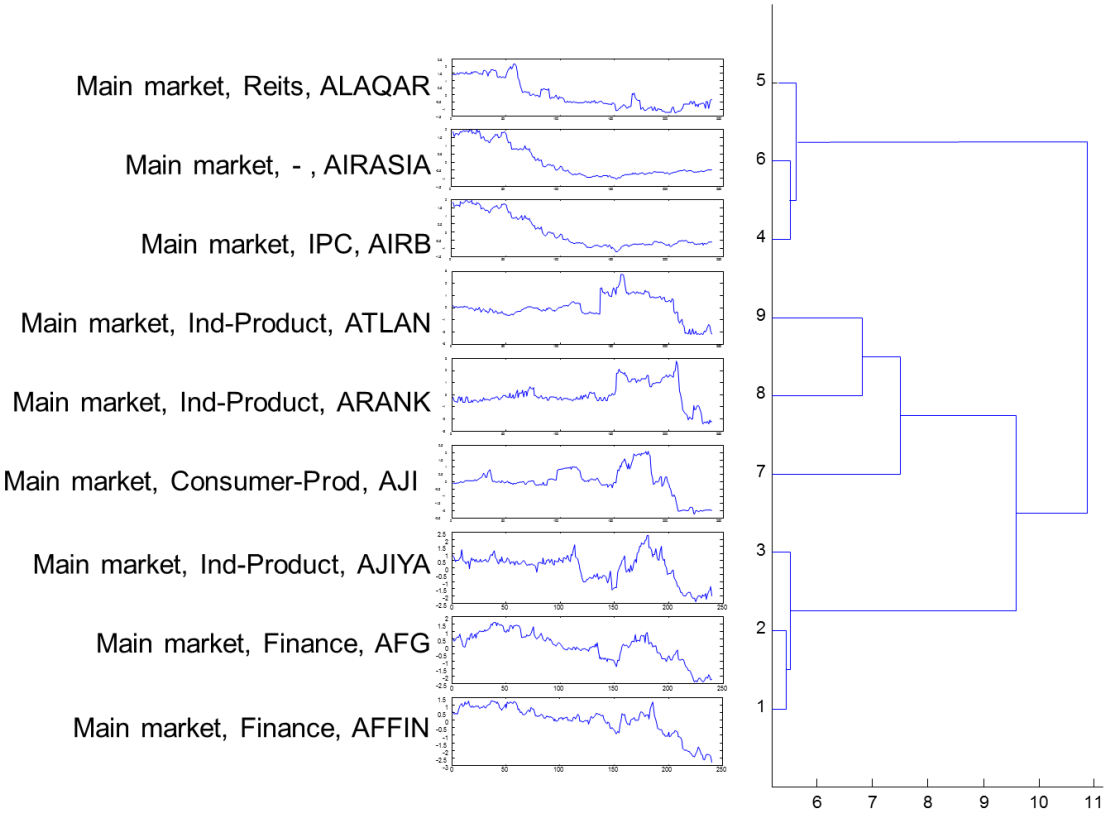


Figure 6.25: A sample of clustering of companies based on their stock exchange in 2010

Figure 6.26 shows the handling of shifts in time-series of stock exchanges related to different companies. For example, two companies (AFFIN and AFG) which are very similar in shape (not in time) are shown in Figure 6.26. It shows that the changes in AFG are very similar to AFFIN but with a time shift.

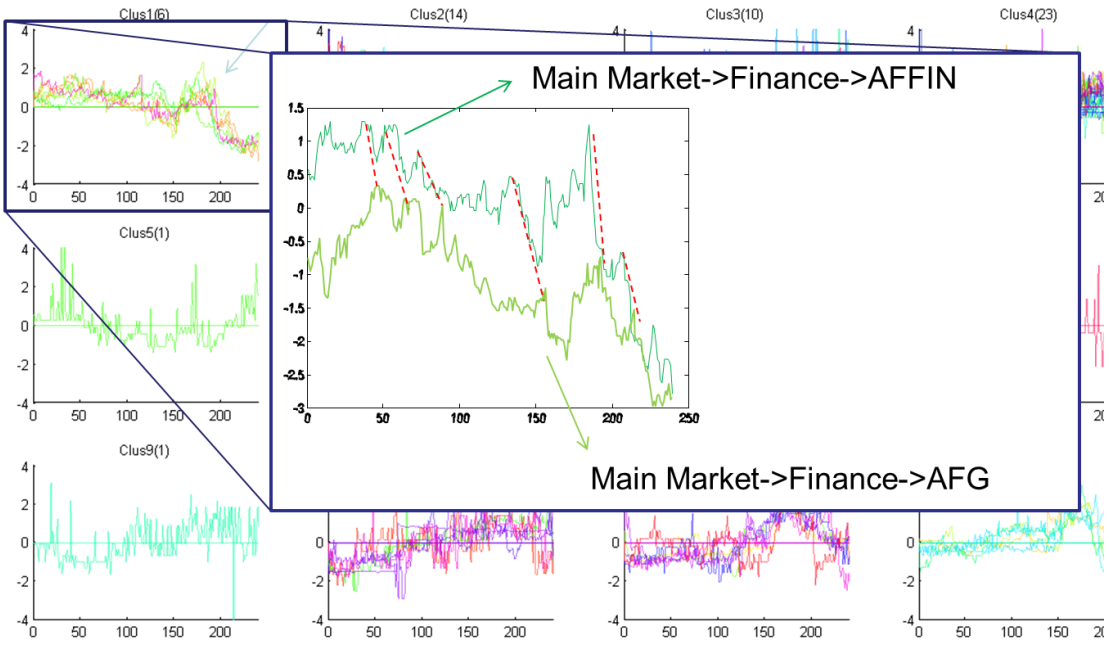


Figure 6.26: Handling shifts in the clustering of time-series of stock exchange of companies in KLSE dataset

As explained in the literature review, to evaluate the results, SSE is used as internal criteria (see Section 2.7.2). Figure 6.27 shows the quality of clustering of KLSE dataset.

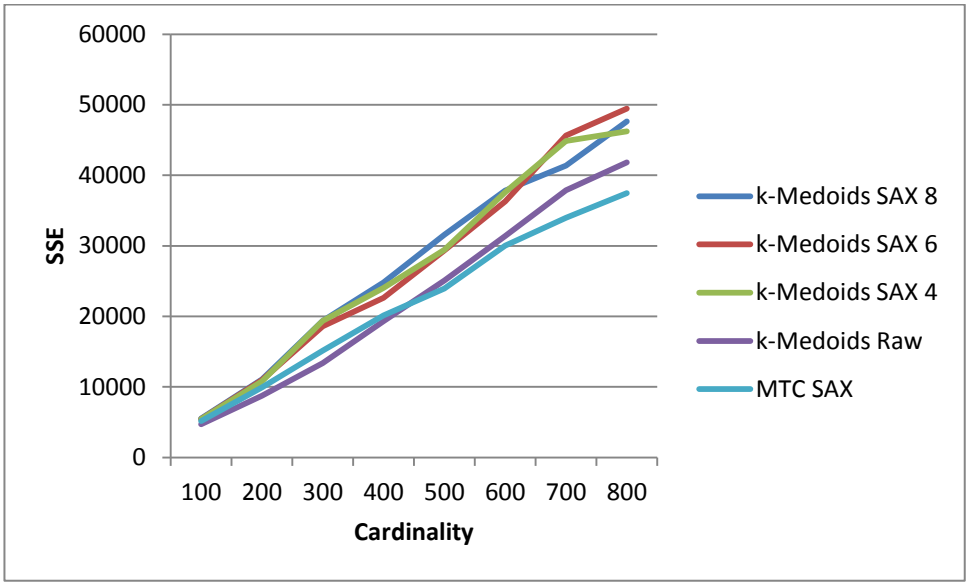


Figure 6.27: Accuracy of MTC in front of different cardinalities of KLSE

In Figure 6.27 different cardinalities of the dataset are chosen for clustering by MTC. As the results show, the quality of MTC is competitive with k-Medoids (using RAW time-series). However, as expected, the trend of quality shows that MTC outperforms in

high cardinality datasets. That is, MTC is more advantageous on large datasets in comparison with small datasets.

6.2.8 Evaluation of IMTC

An interactive version of MTC was introduced in Section 4.7, so called IMTC. IMTC is similar to alternate splitting and merging of MTC. In IMTC, for each iteration, only one pre-cluster is considered to be revised and dispersed. Therefore, the results of the first iterations may be bad and incorrect but gradually it is refined and gets better.

In this experiment, the feasibility of running IMTC in its interactive manner is shown. CBF dataset is utilized to evaluate the accuracy of IMTC. In the first step of IMTC, SAX representation is utilized with APXDIST as a distance measure which was shown that is effective. Five sample cardinalities are chosen for this experiment to show the performance of IMTC on large time-series datasets. The results are depicted in Figure 6.28.

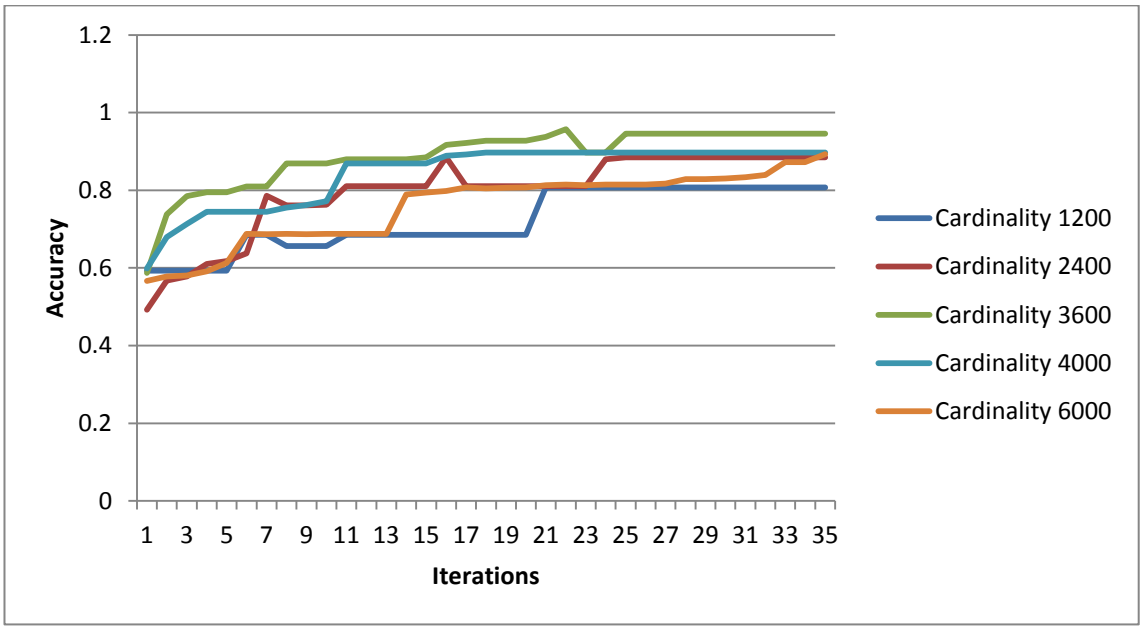


Figure 6.28: Interactive version of Multi-Step Time-Series Clustering (IMTC) on CBF dataset

As the results show, after a few iterations, user obtains better results. As expected, there are two outstanding points about this approach. First, users can see reasonable results

after only a few iterations, i.e., it is guaranteed that user can get better results after first iteration of the algorithm. Second, using IMTC, the final result can even be better than MTC at the expense of time. It is because the clusters are regularly split, their resolution increased, more accurate distance between their objects are calculated, and finally are merged. It can be continued until all low resolution time-series are replaced with high resolution time-series (e.g., to 35 iterations) and the confusion matrix between them are calculated with more accurate distance measure (e.g., DTW). However, the question is why it does not meet the maximum accuracy after a while. Actually, maximum accuracy is not reachable even if the user waits for ever (or may occur randomly), because firstly the error rate of second step hardly gets zero because of data reduction process in this step. Secondly, the quality is calculated in front of ground truth (natural clusters). That is, if even all the distances among all time-series in the dataset are calculated precisely, the quality of clusters is a bit less than 1, because the ground truth is the type of time-series generated syntactically, not the labels gained by clustering of data using a perfect distance measure such as DTW.

To compare IMTC with competitive and similar approaches, IK-Means model (which was discussed in Section 2.8.6) is implemented and used. The proposed algorithm (IK-Means) works interactively and need centroids which should be doubled in each iteration. In IK-Means, k-Means is chosen as the algorithm with PAA as representation for testing the results. CBF dataset is used to show the results of IK-Means. Figure 6.29 shows the accuracy of applying IK-Means on CBF dataset.

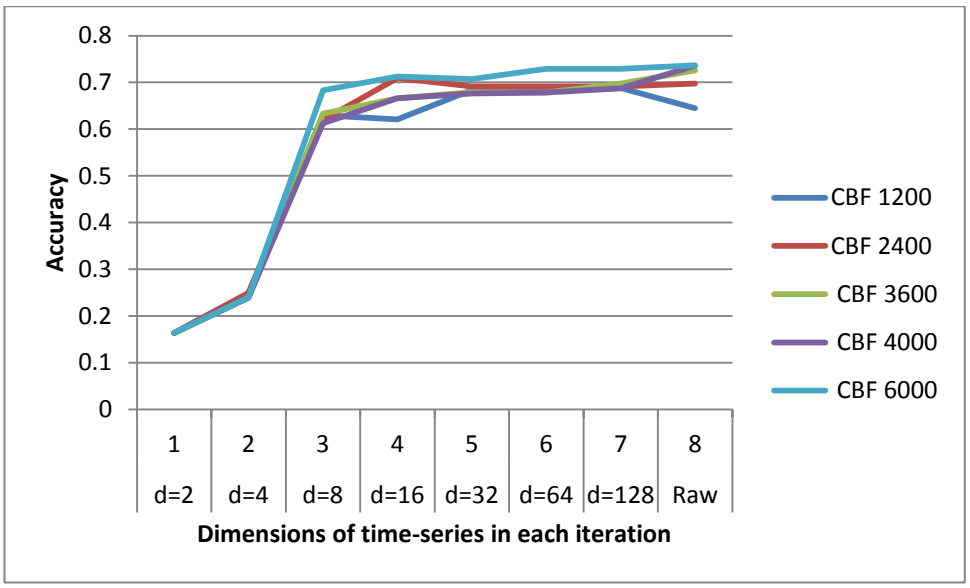


Figure 6.29: Quality of interactive method (IK-Means) for different resolutions of data

As Figure 6.29 shows, in this approach (IK-Means), the number of iterations to the termination point is less than IMTC (Figure 6.28). It is because of the dependency of IK-Means to the length (dimension) of time-series (as discussed in Section 2.8.6). Moreover, because the centroid of k-Means in each iteration (with high resolution time-series) is provided by previous iteration (with lower resolution of time-series), the number of internal iterations in k-Means is decreased for each run. For example, running k-Means on CBF dataset with 2400 objects, it takes around 12 iterations for the lowest resolution time-series (i.e., dimensions=2) and it is decreased to around 4 iterations for the highest resolutions (i.e., dimensions=128). However, as the results (of IK-Means) shows (Figure 6.29), in comparison with IMTC (Figure 6.28), even in the heights resolutions of time-series in IK-Means, maximum quality of IK-Means is less than IMTC. That is, the maximum accuracy of clustering using IK-Means is around 70%, whereas the maximum accuracy is around 90% using MTC. It is because of finding similar time-series in shape in IMTC model which handles the shifts, and also pre-clustering step in IMTC which decreases the effect of outliers (as explained in Section 6.4.2).

6.3 Scalability Evaluation

Scaling and performance are often considered together in data mining. The problem of scalability in the data mining community is not only how to process large sets, but how to do it within a useful timeframe. Analogous to conventional clustering approaches, MTC also simply groups similar time-series. Hence, its efficiency depends on distance measure process, representation process, prototype calculation, and clustering algorithm. In the following subsections, space complexity and time complexity of MTC are analysed.

6.3.1 Space Complexity (Space Utilization)

Space utilization means the amount of memory required to run the algorithm. In the first step of the model, only reduced time-series are stored, so the required space for algorithm depends on the amount of data (time-series dataset) and the compression-ratio. The storage that MTC needs for the first step, is $O((N + k_{PC})r)$ where N is the number of time-series, k_{PC} is the number of clusters (pre-clusters), and r is the length (dimensions) of time-series after SAX transformation. In the second step, the raw time-series (in the worst case) are used as the highest resolution, for refining (purifying). However, because only one cluster of time-series is loaded for refining, and after that its prototype is kept for the next step, the storage that MTC needs, *is* $O((N/k_{PC} + k_{SC})d)$ where the N/k_{PC} is the approximate number of time-series with length d in each cluster, and k_{SC} is the number of sub-clusters (prototypes) in the second step. The third step of clustering is the merging process of prototypes and needs only O(k_{SC}d) storage. Therefore, because all these steps are performed sequentially, and $d<<N$, the highest storage that MTC needs, is $O((N+k_{PC})r)$. It is not a big memory usage, because r can be determined by the user to be very small in comparison to the original length of time-series.

6.3.2 Time Complexity

Time efficiency is the amount of time that is required to process the data. Time efficiency depends on many factors such as the size of data, the speed of the machine which the program is running on that, source code, and compiler. These factors may vary from one machine to another. Hence, only the number of times that the instructions are executed is counted. Therefore, computing time is calculated as follows:

Considering *T(N)* as the computing time of MTC for input of size *N* (the worst-case time complexity of the algorithm), *T(N)* equals to the number of times that the instructions are executed. The overall computational complexity of MTC depends on the amount of time that it requires to construct the first step clusters, to refine the clusters in the second step, and the amount of time it requires to perform the merging of the prototypes in the last step of clustering algorithm. In the following, three steps are analysed sequentially.

Step one: This step includes dimension reduction and pre-clustering. Suppose that the number of time-series is *N*, and the size (length) of time-series is *d*, which changes into r after dimensionality reduction process, where $r < d$. Therefore, first, all time-series are dimensionality reduced. The complexity of dimensionality reduction using SAX is *O(Nd)*. Then, Ek-Modes clustering is used with dimensionality reduced time-series. The amount of time required to make the pre-clusters, depends on the dimensionality of time-series in hand, the distance measure and the number of iterations which it is run "till it is done". Since the dimensionality reduced time-series are used, the complexity of APXDIST distance measurement is linear to the length of the transformed time-series (dimensionality reduced time-series), i.e., *O(r)*. Each iteration in Ek-Modes takes $O(k_{PC}N)$ time to identify the nearest centres and do the new centre computation. Hence, the time complexity of Ek-Modes is $O(Ik_{PC}Nr)$, where k_{PC} is the number of clusters, I is the number of iterations takes to converge, r is the dimension of time-series (the

number of points in dimensionality reduced time-series, $r < d$), and *N* is the number of instances. This complexity is modest, that is, the required time is linear in the number of time-series (*N*), because I is not very big as the most changes happened in the first iterations and the number of iterations which takes to converge is not very high running on dimensionality reduced time-series (as discussed in Section 4.3.4).

Step two: suppose that the number of time-series is *N*, and the number of the constructed pre-clusters from the pre-clustering step is k_{PC}. This value can be taken either equal to *k* as class labels (defined by user) or equal to $\sqrt{\frac{N}{2}}$ (Han & Kamber, 2011) , where $k_{PC} \ll N$). Complexity of PCS in the best case is same as the CAST algorithm which is a bit more than O($N.logN$) (Ben-Dor et al., 1999) and in the worst case is $O(N^2)$ (Bellaachia et al., 2002). Considering the complexity of Euclidean distance as $O(d)$, the complexity of PCS is $O(\frac{N^2}{{k_{PC}}^2}d)$. Since there are k_{PC} such partitions, the overall complexity becomes $O(\frac{N^2}{k_{PC}}\mathrm{d})$. After using PCS to partition the pre-clusters (generated clusters in the first step), k_{SC} sub-clusters are generated as $k_{SC} = k_{PC} + q$, where $1 < q \ll N$. Usually q is a small number because indicates the number of incorrect members or outliers in the cluster.

Step three: Suppose k_{SC} is the number of sub-clusters in the second step, which reduce the size of dataset by the reduction-factor of R_f which is defined based on reduction-rate (see 5.3.2.1 for definition) as:

$$R_f = 1 - R_{rate} \qquad 6.2$$

Then, if the final clustering scheme is for example a hierarchical clustering (e.g., average linkage), complexity is $O(N^2\log(N))$ (Hartigan, 1975; Murtagh, 1985) where *N* is the number of input time-series, and $O(d^2)$ for DTW between each pair of time-series. Although there are some "lower bounds of DTW" and "DTW with constraints"

which are relatively cheaper than complete DTW in time complexity, i.e., $O(d)$, as explained in Section 4.7, in the worst case its complexity is $O(d^2)$, where d equals to the length of time-series (Keogh & Ratanamahatana, 2004). Therefore, total computation of merging is $O(N^2 \log(N).d^2)$.

Overall complexity: Substituting $N.R_f$ (i.e., number of prototypes) in the equation of the third step, the overall time required for three steps is

$$O(MTC) = O\left(N^2.{R_f}^2 \log(NR_f).d^2 + \frac{N^2}{k_{PC}}.\mathrm{d} + \mathrm{I}k_{PC}\mathrm{Nr} + \mathrm{Nd}\right) \quad 6.3$$

Given the complexity of MTC (using the hierarchical scheme) as $O(MTC)$, and conventional hierarchical algorithm which is in order of $O(N^2 \log(N).d^2)$ as $O(Hier)$, for comparing MTC with hierarchical algorithm, the ratio of complexity is shown by:

$$Ratio = \frac{O(MTC)}{O(Hier)} \quad 6.4$$

Therefore, ratio of complexity of MTC on hierarchical algorithm is calculated as:

$$Ratio = \frac{{R_f}^2 \log(N.R_f)}{\log(N)} + \frac{1}{k_{PC}\log(N)d} + \frac{\mathrm{I}k_{PC}\mathrm{r}}{\mathrm{N}\log(N).d^2} + \frac{1}{\mathrm{N}\log(N).d} \quad 6.5$$

because $R_f < 1$ then $\log(N.R_f) < \log(N)$, cancelling out the $\log(N)$, the ratio of complexity of MTC on hierarchy algorithm becomes:

$$Ratio = {R_f}^2 + \frac{1}{k_{PC}\log(N)} + \frac{\mathrm{I}k_{PC}\mathrm{r}}{\mathrm{N}\log(N).d^2} + \frac{1}{\mathrm{N}\log(N).d} \quad 6.6$$

Considering for example the dimension reduction of r=d/4 (i.e., compression-ratio=4, e.g., SAX4), and $k_{PC} \ll \mathrm{N}$, for example $k_{PC} = \sqrt{\frac{N}{2}}$, then

$$Ratio = {R_f}^2 + \frac{1}{\sqrt{\frac{N}{2}}\log(N)} + \frac{\mathrm{I}}{4\sqrt{2N}\log(N)d} + \frac{1}{\mathrm{N}\log(N)d} \quad 6.7$$

Because $R_f < 1$ (based on Equation 6.2) and I is small (as explained in Section 4.3.4), then $Ratio < 1$, that is, $Ratio$ is always less than 1. Therefore, it can be concluded that the complexity of MTC is better than hierarchical algorithm.

6.4 Sensitivity Evaluation

In this subsection, a sensitivity analysis for MTC is performed with respect to the parameters. The parameters are the initialization of clustering and resolution of time-series in the first step. The synthetic datasets and all datasets in UCR are used again to evaluate the sensitivity of algorithm in different conditions. In the following, at first the effect of random initialization in the first step is discussed.

6.4.1 Effect of Random Initialization

The quality of the MTC clustering model relatively depends on the quality of pre-clusters made in the first step. However, in the first step, Ek-Modes are used where quality of the results depends on the choices of the initial centres, similar to other partitioning algorithms. That is, MTC uses a random initialization in the first step. Therefore, since it may encounter local minimum, the best results are not generated in the first step (the process of making the pre-clusters). Hence, the clustering results may not be stable. To test the stability of clustering results, MTC is run on 13 different cardinalities of CBF and CC dataset. MTC is performed numerous times and its quality is reported to evaluate the clustering model. Some statistical values such as minimum, maximum, mean, median or standard deviation is used to determine the actual quality of the clustering approach. MTC is carried out 100 times with random initializations. To remove the effect of SAX parameters, the same compression-ratio of SAX is considered in this experiment. The results of clustering are depicted in three boxplots in Figure 6.30, Figure 6.31 and Figure 6.32.

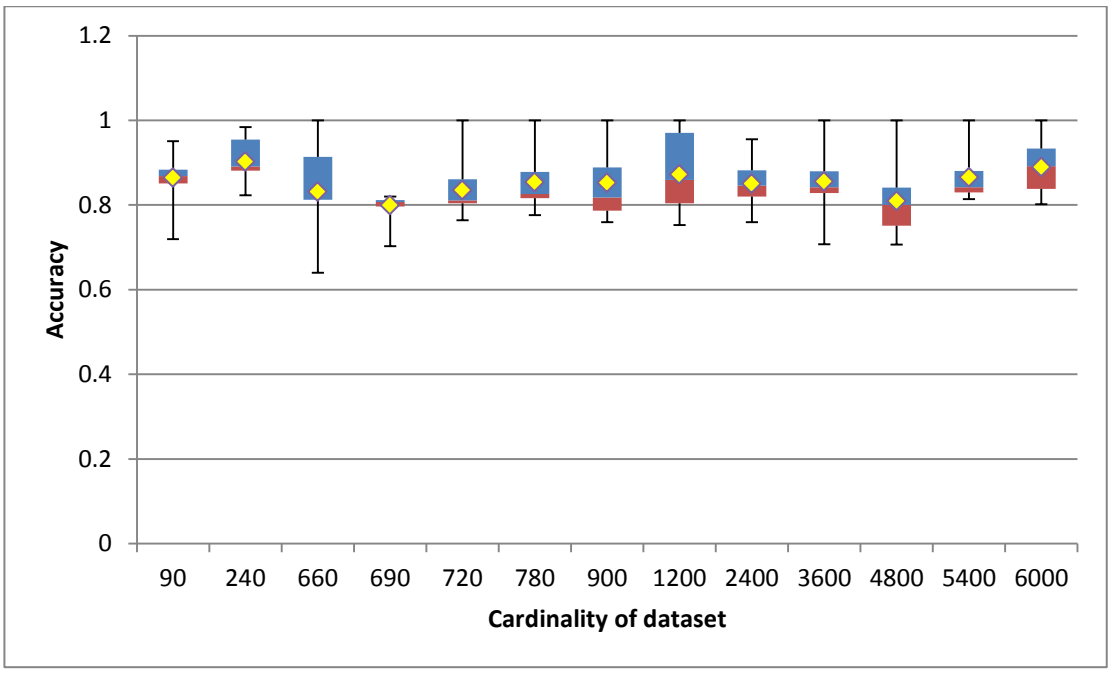


Figure 6.30: Quality of clustering of CBF using MTC (k-Medoids scheme) with random initialization

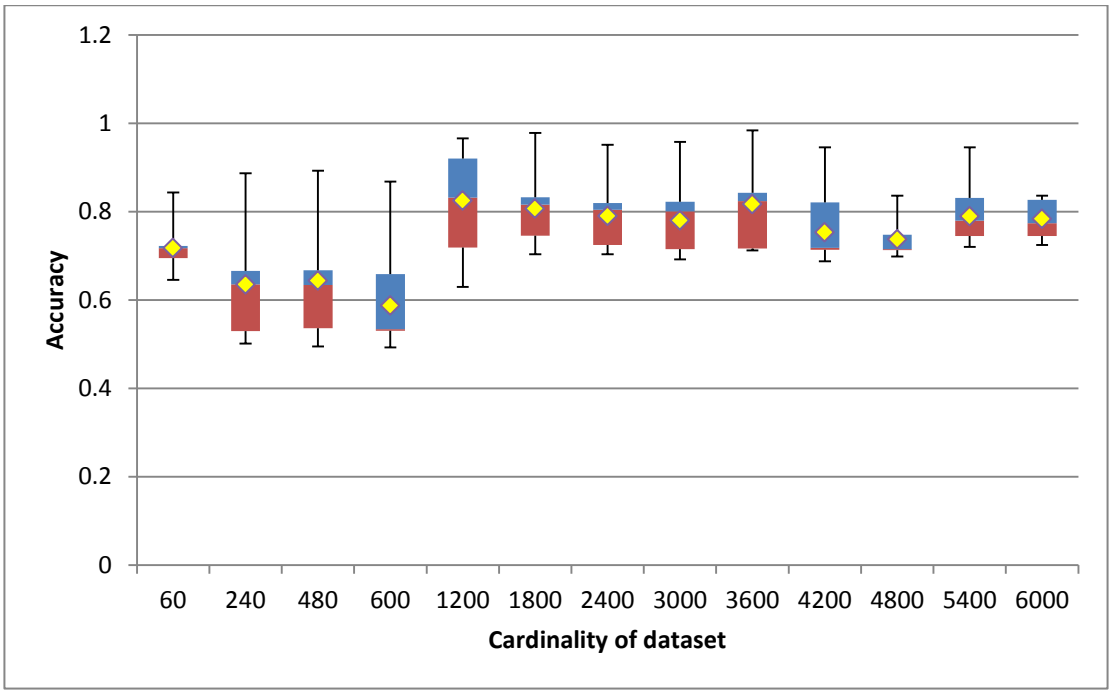


Figure 6.31: Quality of clustering of CC using MTC (k-Medoids scheme) with random initialization

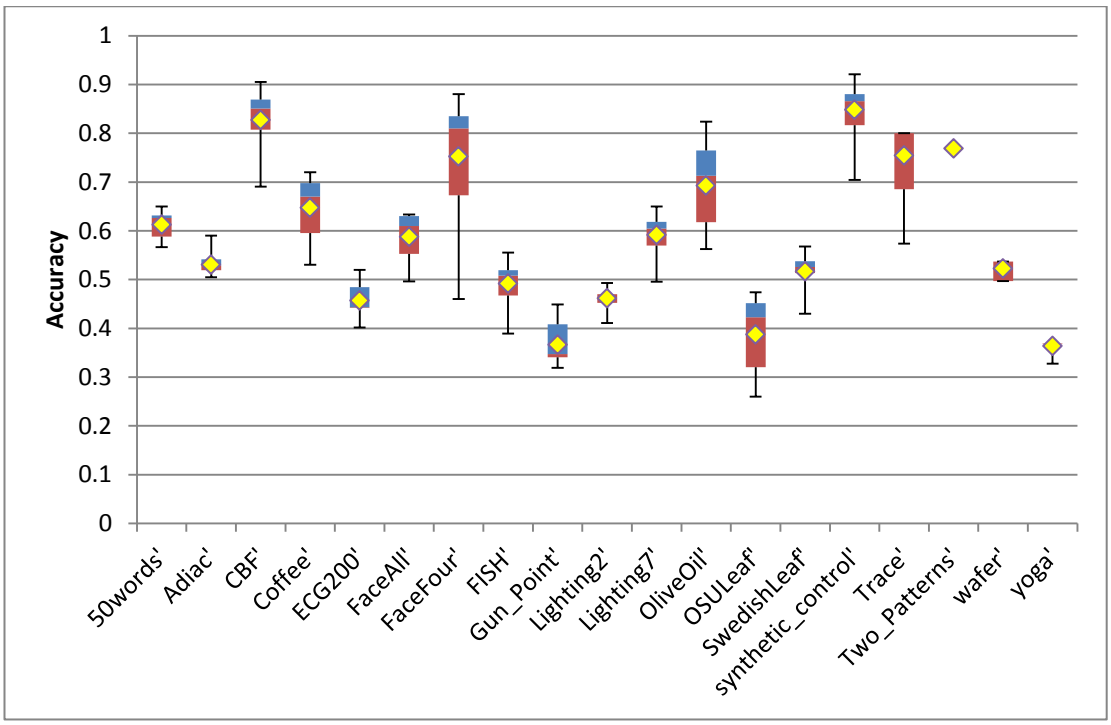


Figure 6.32: Sensitivity of MTC on UCR dataset with random initialization (SAX4)

The plots clearly suggest that some datasets are sensitive to random initialization of centroids in the first step of clustering, and some of them are not sensitive to the random initialization. However, in most of the cases the quality is not very critical to random initialization. It is mostly because of using the lower dimension of time-series in the first step, because it results in not falling in the trap of local minima which is consistent with the findings of Ding et al. (2002). Moreover, a deep-looking at different datasets in UCR repository (see Figure 6.32), it is understood that for more datasets, MTC is stable and robust to random initialization, especially in large datasets (datasets with high cardinalities) for example Wafer and Yoga datasets. Similarly, the graph related to CC also shows that this variation gradually declines in high cardinalities of the dataset. As a result, random initialization of first level may a bit affects the final quality in some datasets which is unavoidable. From another point of view, usually clustering algorithms are performed many times to get the best results; therefore, it is solved easily with a few running. Moreover, generating several structures of data is not very bad in the case that clustering is performed for the purpose of determining the structure.

6.4.2 Effect of SAX Parameters

As mentioned, the Ek-Modes algorithm is used with time-series represented by SAX in the first step of the proposed model. For using SAX as representation, some parameters should be set up. One of them is compression-ratio. The effect of different parameters on the first step of MTC was investigated in Section 5.3.1.1. In the following, the effect of compression-ratio is investigated on MTC. To remove the effect of random initialization in this experiment, the maximum accuracy for each compression-ratio is considered. The results of different compression-ratio for time-series in the UCR dataset, and CBF dataset are illustrated in Figure 6.33 and Figure 6.34 respectively.

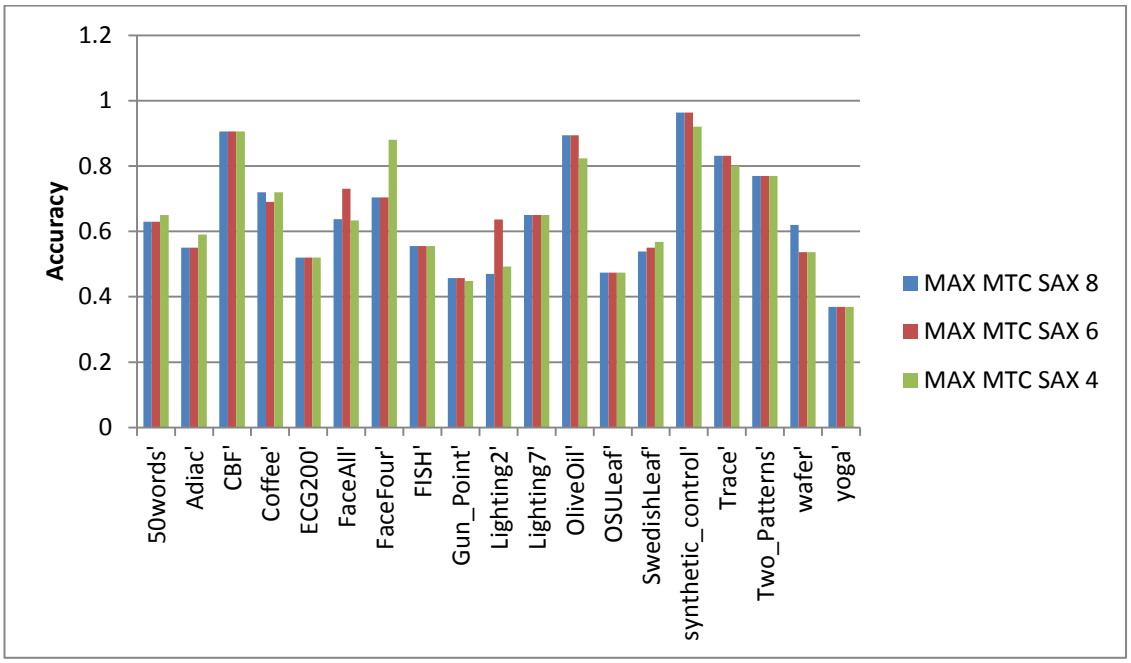


Figure 6.33: Quality of clustering of UCR dataset using MTC with three different compression-ratio of SAX

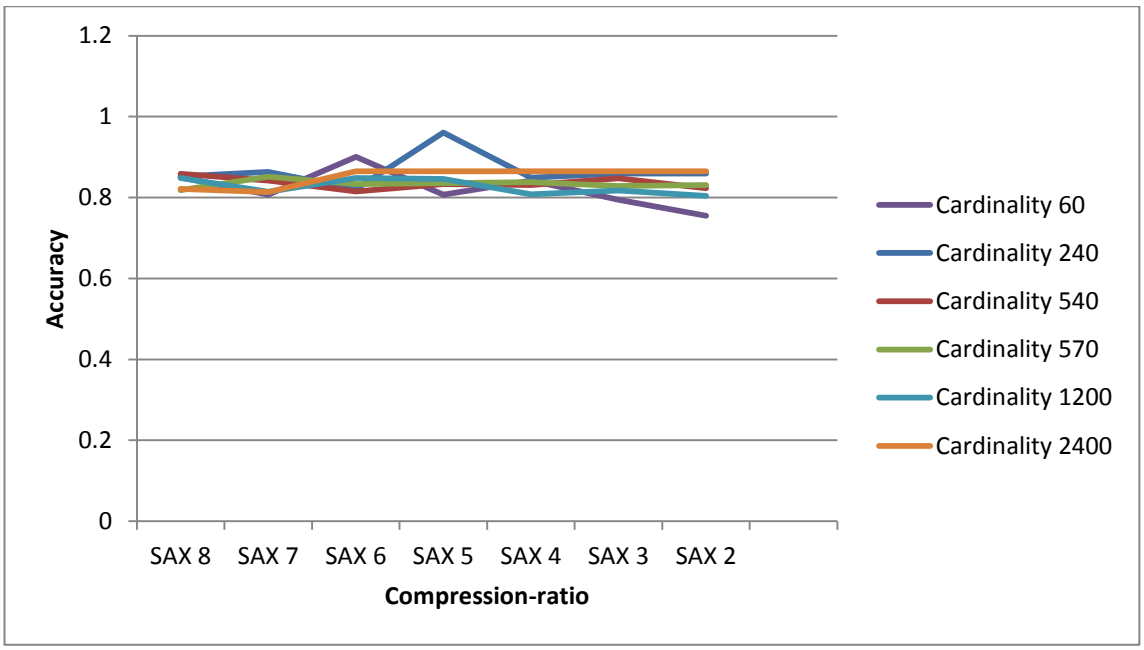


Figure 6.34: Quality of clustering of various cardinalities of CBF using MTC with seven different compression-ratio of SAX

The results in both experiments (Figure 6.33 and Figure 6.34) show that the resolution of time series which are used in the first step of MTC, does not have a very critical effect on accuracy. Although, clustering on a specific dataset represented by a specific resolution may construct more precise results, or slightly worse than our expectation, in general they are very close. For example, Figure 6.20 shows that the average quality of CBF is around 0.8, and it is relatively stable with increasing the resolution. It demonstrates that though the choice of the SAX parameter affects the final results, it does not have a very significant impact on the clustering quality. Surprisingly, it rather gets worse in some datasets, for instance it is seen that in some higher resolutions of time-series (e.g., SAX2), the quality of a clustering approach may decrease as it is seen in Figure 6.34. It is the effect of the presence of noise and outliers in higher resolutions.

6.5 Chapter Summary

In this chapter, the results obtained by applying MTC on different datasets were evaluated visually and objectively. It was visually shown that clustering can be applied on a large time-series dataset to generate a dendrogram of meaningful and accurate

clusters using MTC. Different evaluation methods were used to evaluate MTC extensively. MTC outperforms other conventional clustering algorithms experimenting on various datasets. It was shown that using pre-clustering and prototyping, accurate clusters are made. The results were compared with other multi-step methods to show its strength. The researcher's proposed method can achieve best clustering results compared against several other popular time-series clustering methods. Moreover, MTC was applied on large time-series and the obtained result was discussed. It was revealed that user does not require using very low dimensional time-series for clustering of large datasets; instead, the clustering can be applied on smaller sets of high dimension time-series by prototyping process. That is, the cost of using representatives is much less than dimension reduction in terms of accuracy. Subsequently, MTC was applied on two datasets, where the class labels were not available. The results show that MTC can find the clusters of time-series which are similar in shape especially in large datasets. As an extension of MTC, the interactive version of MTC (IMTC) was utilized on some datasets to show the scalability of the algorithm. It was shown that reasonable results are obtained after only a few iterations, and the quality of the final result can even be better than MTC at the expense of time. Space and time complexity of the proposed model were computed. The computation shows that the execution time of the model is acceptable. Finally, the sensitivity of MTC was evaluated versus random initialization and parameter setting.

7.0 CONCLUSION

7.1 Introduction

This study demonstrates the utility of a multi-step clustering (MTC) model for time-series clustering. Instead of clustering dimensionality reduced time-series data (or naively raw data) using traditional clustering approaches, MTC clusters the time-series data in a multi-step manner that results in substantially higher accuracy and moderate cost in time complexity. This chapter concludes the research on providing a support environment for novice researchers working on time-series clustering. The summary of findings, limitations of the research and recommendations are given in the following subsections.

7.2 Summary of Results and Findings

First, the main reasons for low quality in time-series clustering are stated based on all the findings discussed in the literature review (Chapter 2) and the evaluation chapter (Chapter 6). The reasons are drawn by answering the following questions:

What is the main reason for low quality time-series clustering?

1. Use of dimensionality reduction approaches through the whole clustering process (although it is used to remedy the high complexity of distance measuring of time-series).
2. Use of insufficiently accurate or inappropriate distance measures to calculate dissimilarity between the time-series (mostly because of the high complexity of accurate metrics).
3. Use of conventional static clustering algorithms that are not suitable for time-series data (mostly because of the high complexity of state-of-art algorithms).

The proposed models in this study can improve the quality of clustering, while maintaining clustering efficiency using the principles of pre-clustering and cluster-

revising. In Chapter 5, the effect of dimension reduction on clustering was described. Dimensionality reduction can lead to data oversight, and different results per compression. The factors that made MTC accurate are shown by discussion on the different steps of the model. Moreover, using evaluation methods described in Chapter 6, MTC has been proven to work much better than other algorithms. The results were compared with the ground truth as natural structure of data to show meaningfulness of clustering. Finally, MTC has been proven practical and usable by real-world samples. The points that made MTC superior to other algorithms are briefly answered with the following question: Why is MTC more accurate than other algorithms?

1. Time-series are clustered as low-resolution data in the first step. Therefore, the algorithm efficiently finds the pre-clusters, avoids local minima, and is robust to the outlier time-series.
2. Given that a raw time-series or a very high-resolution time-series (in the case of noisy time-series) is used in the second step of MTC (instead of low-resolution time-series in the first step), the accuracy of the final results is very high.
3. An accurate distance metric (DTW) can be utilized in this process that handles all shifts inside the time-series, resulting in very accurate clustering, because of the multi-step manner of the proposed model (given the high complexity of DTW, its use is infeasible on large datasets in other methods).
4. Using prototypes prepared in the middle step of MTC, the user has a chance to choose his final clustering scheme based on the domain and create more meaningful clusters, i.e., an algorithm capable of making arbitrary shape clusters.

Finally, the researcher is able to deduce the following conclusions:

1. Dimensionality reduction is a common approach for time-series clustering; however, it is not always a perfect approach (in clustering) to remedy the cost of

the high complexity of accurate dissimilarity measurement among time-series. Instead, a more sophisticated clustering algorithm such as MTC can be utilized to cluster high-dimensional time-series without overlooking and losing accuracy.

2. Although all time-series in each sub-cluster are represented by only a few representative time-series, experiments show that this approximation (approximated sub-clusters) has less destructive effect on the accuracy of the final clusters compared with the use of approximate time-series.
3. The proposed model (MTC) is a proper approach for data which are large in size and have shift and noise in its time-series.

7.3 Achievement of the Objectives

The following are the objectives of the research:

1. To propose and develop a new clustering model that accepts large raw time-series data as input, and generates accurate clusters without violating the time execution. In order to fulfill this objective, the following methods need to be developed.
    a. Developing a distance measure for similarity calculation.
    b. Developing a clustering approach for approximate clustering of the time-series data transformed to symbolic representation.
    c. Developing a method to dynamically split the pre-clusters into pure clusters.
2. To extend the proposed model, enabling it to run interactively.
3. To evaluate the capability of the proposed methods in improving the accuracy.

A multi-step clustering approach (MTC) was proposed to accurately cluster the time-series to achieve the first objective. The model was designed, implemented and, its

improvement in accuracy was shown by an extensive evaluation on various domains. The approaches used and the outcomes are as follows:

- A pre-clustering method was proposed to rapidly obtain the approximate clusters (of time-series represented by SAX). The method highly reduces the search area for accurate distance measurement in the purifying step. A distance measure method (APXDIST) was then developed to calculate the distance measure between the time-series represented by the SAX representation to create the pre-clusters. The effectiveness of APXDIST on dimensionality reduced data was evaluated in Section 5.3.1.2
- An extended k-Modes algorithm (Ek-Modes) was developed to create the pre-clusters. The results showed that the combination of the APXDIST and Ek-Modes methods generates higher accuracy of pre-clusters for MTC (and so IMTC) than other approaches (Section 5.3.1.3).
- A purifying method (PCS) was proposed to checking the pre-clusters and splitting them into more pure clusters based on the ED distance measure. This process provides very similar groups of time-series that can be presented by one/a few prototypes precisely (Section 5.3.2).

The proposed model is extended to achieve the second objective. Considering the desirability of an interactive clustering model, IMTC was proposed to generate better results over time. Exploiting the multi-step property of MTC and the concept of "split and merge", MTC was extended to IMTC. IMTC is very practical for users who tend to see a prototype of clusters in each instance. The results showed that the accuracy of clusters improved even more when IMTC was used over MTC, but at the expense of time.

An extensive evaluation was performed to show the superiority of the proposed model to achieve the third objective. The following concepts were considered to show that the

proposed model is not biased towards the dataset, size of data, parameter setting, and accuracy measurement.

- The model was applied to different datasets from various domains and different cardinalities (taken from the largest published time-series repository in the world).
- Different cardinalities of synthetic datasets were generated to show the changes in the accuracy of clusters.
- Nine most-used indexing measures in the literature were utilized to calculate the accuracy of the clustering structure.
- Various ranges of parameters were used to show the sensitivity of the system.

7.4 Contributions

There are many approaches to the clustering of time-series datasets; however, most focus on speeding up the clustering using dimensionality reduction of time-series and utilizing conventional clustering algorithms, which leads to low quality when dealing with large amounts of time-series data. The common problem of low quality in these outputs was identified. Then, focusing on the components of time-series clustering, a new multi-step clustering model was proposed for the accurate clustering of time-series datasets. Accordingly, the contributions of this thesis research are outlined as follows:

- A new multi-step model for time-series clustering (MTC)
- A new similarity measure compatible with SAX (APXDIST)
- Finding and customizing the best clustering algorithm compatible with the symbolic representation of time-series (Ek-Modes)
- A new approach for splitting approximate clusters to pure sub-clusters (PCS)
- A new model for interactive clustering of time-series data (IMTC)

A new clustering model (MTC): The main objective of this study is to find accurate clusters. Accuracy means very pure and meaningful clusters, which can explain the structure of data precisely. It is considered a hierarchy of data that is understandable for human beings. The main contribution of this study is in proposing a new clustering model to achieve this objective. This model (MTC) can find the accurate clusters of large time-series that are similar in shape and can be represented visually. To the best of the researcher's knowledge, MTC is the first model based on the large time-series datasets to obtain meaningful clustering results. This method is performed in multiple steps and has the following features:

- Very accurate and interpretable: Using high-resolution time-series and high-precise distance measure (DTW), clusters are made accurately. This model shows the structure of data (which are similar in time and shape) hierarchically, which is more usable and understandable for users.
- Practical: Using dimensionality reduced data in the first step of MTC, data are clustered approximately (this action is necessary to reduces memory requirements because all raw time-series cannot fit in the main memory). Large datasets are then clustered with less distance calculation compared with conventional algorithms by exploiting approximated clusters.
- Flexible for arbitrary shape clusters: In the last step of MTC, prototypes of pure sub-clusters are used. As a result, the number of time-series in the third step is much less than the original data, enabling MTC to work with all types of clustering algorithms to produce final clusters. This method is also desirable for different type of clusters (clusters of different types, clusters of various density, and non-globular clusters). As a result, the user can choose the final clustering algorithm, providing him the chance to utilize the best algorithm based on the domain on hand and on the shape of desired clusters.

- Visualization: Given that a user has the access to the middle-step results (prototypes), which are very small (in comparison with the whole data), the results can be shown visually (e.g., by dendrogram). The user can then choose a part of the data (e.g., by removing some outliers) for final clustering, making the clustering results more meaningful and practical.

A new distance measure (APXDIST): In the first step of the time-series clustering, a symbolic approximation approach was used to find approximate clusters. A similarity measure, APXDIST, which is more accurate for finding dissimilarity among SAX time-series than existing approaches, was developed to calculate the similarity of approximated time-series. The proposed new distance measure method answers the first research question of this study (Q1) and provides new knowledge to researchers on how to find accurate approximate clusters.

A new algorithm for clustering of approximated time-series (Ek-Modes): Although k-Modes is not a new algorithm, its compatibility with categorical objects was used in this study for the first time to develop a clustering algorithm (Ek-Modes) to cluster the approximated time-series. The evaluation results indicated higher quality compared with competitive approaches, answering the second question of this research (Q2). The significance of this finding is its application as a clustering approach for standalone systems that work with symbolic representations of time-series data (e.g., SAX).

A new post-processing approach for approximated clusters (PCS): A research question was posed in this study (Q3) as "How can constructed clusters be revised as a post-clustering action to achieve better results?" A new approach, called PCS, for splitting approximate clusters into pure sub-clusters was proposed in this study. PCS is a post-processing approach to create sub-clusters (from pre-clusters) sequentially with a dynamic affinity threshold. This approach is important from two different points of view. First, it can be considered as a post-processing method for other models that work

with multi-resolution time-series or multi-precise distance measures. Second, it is a dynamic approach that does not use any parameter, which is very desirable in clustering.

A new model for interactive clustering of time-series (IMTC): Designing the MTC model as a multi-step approach provided the capability of extending the model to be proposed as an interactive clustering approach (Q4). Clustering can be carried out interactively using this model. That is, the user can stop the process, check the results, and let the process be continued (if the results are not satisfactory) or be terminated (if it met the needed accuracy). Moreover, this method provides the user with the chance to stop the process in the initial iterations and change the parameters of clustering if the results are wrong. This feature is very important for data mining approaches, especially for large datasets.

An extensive evaluation: To the best of the researcher's knowledge, there are no published results for different clustering algorithms on various datasets. This study is the first to test these ranges of datasets using different clustering approaches. Moreover, the effect of a representation method on time-series clustering versus a raw time-series was investigated for both partitioning and hierarchical clustering. The method provides a good range of accuracies for researchers who are working on clustering of time-series to compare their results (Q5).

7.5 Limitations of the Current Study

Some limitations that need to be considered are as follows:

1) This research only managed to focus on proposing methods for short and moderate length of time-series because of the time constraint in achieving the main objective. For the long time-series, the same problem (low accuracy) also exists, which is worth carrying out as further research.

2) The model was not evaluated versus different dimensionality reduction methods, though, SAX representation was taken as the best representation method in the literature. However, more testing and analysis are required to determine the optimum representation method for the model.
3) Given that MTC is running as interactive clustering, the results are sometimes not stable. This phenomenon is also observed when running MTC on different cardinalities of CBF and CC dataset. If the clustering of time-series is seen as a part of more complex task (pre-processing), then it may cause problems and needs to be addressed by running a few times and choosing the best answer. However, if time-series clustering is seen as a tool for human understanding of a dataset, it may be beneficial to look at more than one clustering structure. Moreover, an unstable algorithm usually discovers interesting structures.

7.6 Recommendation and Future directions

The researcher strongly believes that the proposed model in this study highly improves the accuracy of existing approaches in time-series clustering. It is the researcher hope that this work will be a good motivation for researchers who need more accurate and meaningful clustering in other applications, including image categorization, pattern recognition, and fraud detection, etc. However, there are still a few aspects that need further studies. The following are recommendations for future studies, suggested based on the limitations of this research area.

1. In order to represent a cluster of time-series, prototype/s is needed to represent the cluster. Find an accurate approach to represent a cluster made by elastic distance measure (e.g., DTW) is also needed. As mentioned in the literature review, certain approaches to the prototyping of clusters of time-series are available. However, most of these approaches are quite expensive or inaccurate.

As a result, developing an efficient and effective approach to define a prototype for time-series clusters can be significant.

2. Accuracy evaluation of the clustering structure is still an open problem in the data mining community. Although several indices (internal and external) are available for the evaluation of clustering accuracy, visualization is usually used as last resort. Visualization of the clusters for static objects is relatively easy; however, it is not a trivial task to visualize time-series clusters, especially for large datasets. Therefore, there is a need for an accurate evaluation of time-series clustering.
3. The accuracy of clustering is tightly regulated by the distance measure between time-series. However, the accurate measurement of similarity between time-series using existing distance metrics is expensive in terms of execution time. Approximating the measurement is required in time-series clustering.

REFERENCES

Aach, J., & Church, G. M. (2001). Aligning gene expression time series with time warping algorithms. *Bioinformatics*, *17*(6), 495.

Abdulla, W., & Chow, D. (2003). Cross-words reference template for DTW-based speech recognition systems. *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region* (Vol. 4, pp. 1576–1579). IEEE.

Abonyi, J., Feil, B., Nemeth, S., & Arva, P. (2005). Principal component analysis based time series segmentation-application to hierarchical clustering for multivariate process data. *IEEE international conference on computational cybernetics* (pp. 29–31).

Aghabozorgi, S., Saybani, M. R., & Wah, T. Y. (2012). Incremental Clustering of Time-series by Fuzzy Clustering. *Journal of Information Science and Engineering*, *28*(4), 671–688.

Aghabozorgi, S., Wah, T. Y., Amini, A., & Saybani, M. R. (2011). A new approach to present prototypes in clustering of time series. *The 7th International Conference of Data Mining* (Vol. 28, pp. 214–220). Las Vegas, USA.

Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases. *Foundations of Data Organization and Algorithms*, *46*, 69–84.

Alcock, R., & Manolopoulos, Y. (1999). Time-series similarity queries employing a feature-based approach. *7th Hellenic conference on informatics, Ioannina, Greece* (pp. 1–9).

Alon, J., & Sclaroff, S. (2003). Discovering clusters in motion time-series data. *Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 375–381). IEEE Comput. Soc.

Amigó, E., Gonzalo, J., Artiles, J., & Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, *12*(4), 461–486.

Andreopoulos, B., An, A., & Wang, X. (2005). Clustering the internet topology at multiple layers. *WSEAS Transactions on Information Science and Applications*, *2*(10), 1625–1634.

Andreopoulos, B., An, A., & Wang, X. (2009). A roadmap of clustering algorithms: finding a match for a biomedical application. *Briefings in Bioinformatics*, *10*(3), 297–314. doi:10.1093/bib/bbn058

Ankerst, M., Breunig, M., & Kriegel, H. (1999). OPTICS: Ordering points to identify the clustering structure. *ACM SIGMOD Record*, *28*(2), 40–60.

Antunes, C., & Oliveira, A. L. (2001). Temporal data mining: An overview. *KDD Workshop on Temporal Data Mining* (pp. 1–13). Citeseer.

Apostolico, A., Bock, M. E., & Lonardi, S. (2003). Monotony of surprise and large-scale quest for unusual words. *Journal of Computational Biology*, *10*(3-4), 283–311.

Aref, W. G., Elfeky, M. G., & Elmagarmid, A. K. (2004). Incremental, online, and merge mining of partial periodic patterns in time-series databases. *Transactions onKnowledge and Data Engineering*, *16*(3), 332–342.

Aßfalg, J., Kriegel, H. P., Kroger, P., Kunath, P., Pryakhin, A., & Renz, M. (2008). T-Time: threshold-based data mining on time series. *24th International Conference on Data Engineering, 2008. ICDE 2008. IEEE* (pp. 1620–1623). Ieee.

Aßfalg, J., Kriegel, H. P., Kröger, P., Kunath, P., Pryakhin, A., & Renz, M. (2006). Similarity search on time series based on threshold queries. *Advances in Database Technology-EDBT 2006*, 276–294.

Bagnall, A. J., & Janacek, G. (2005). Clustering Time Series with Clipped Data. *Machine Learning*, *58*(2), 151–178.

Bagnall, A. J., Janacek, G., De la Iglesia, B., & Zhang, M. (2003). Clustering time series from mixture polynomial models with discretised data. *Proceedings of the second Australasian Data Mining Workshop* (pp. 105–120).

Bagnall, A. J., Ratanamahatana, C., Keogh, E., Lonardi, S., & Janacek, G. (2006). A Bit Level Representation for Time Series Data Mining with Shape Based Similarity. *Data Mining and Knowledge Discovery*, *13*(1), 11–40.

Banerjee, A., & Ghosh, J. (2001). Clickstream clustering using weighted longest common subsequences. *Proc. of the Workshop on Web Mining, SIAM Conference on Data Mining* (pp. 33–40). Citeseer.

Bao, D. (2007). A generalized model for financial time series representation and prediction. *Applied Intelligence*, *29*(1), 1–11.

Bao, D., & Yang, Z. (2008). Intelligent stock trading system by turning point confirming and probabilistic reasoning. *Expert Systems with Applications*, *34*(1), 620–627.

Bar-Joseph, Z., Gerber, G., Gifford, D. K., Jaakkola, T. S., & Simon, I. (2002). A new approach to analyzing gene expression time series data. *Proceedings of the sixth annual international conference on Computational biology* (pp. 39–48). ACM.

Bellaachia, A., Portnoy, D., Chen, Y., & Elkahloun, A. G. (2002). E-CAST: a data mining algorithm for gene expression data. *Workshop on Data Mining in Bioinformatics* (pp. 49–54). Citeseer.

Ben-Dor, A., Shamir, R., & Yakhini, Z. (1999). Clustering gene expression patterns. *Journal of computational biology*, *6*(3-4), 281–297.

Bergroth, L., & Hakonen, H. (2000). A survey of longest common subsequence algorithms. *Seventh International Symposium on String Processing and Information Retrieval, 2000. SPIRE 2000. Proceedings.* (pp. 39–48).

Beringer, J., & Hullermeier, E. (2006). Online clustering of parallel data streams. *Data & Knowledge Engineering*, *58*(2), 180–204.

Berkhin, P. (2006). A survey of clustering data mining techniques. *Grouping multidimensional data*, *2*(1), 25–71.

Berndt, D., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *AAAI94 workshop on knowledge discovery in databases* (pp. 359–370).

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

Bicego, M., Murino, V., & Figueiredo, M. (2003). Similarity-based clustering of sequences using hidden Markov models. *Machine learning and data mining in pattern recognition*, *2734*(1), 95–104.

Biernacki, C., Celeux, G., & Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(7), 719–725.

Bingham, E. (2001). Random projection in dimensionality reduction: applications to image and text data. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 245–250).

Bozkaya, T., Yazdani, N., & Özsoyoğlu, M. (1997). Matching and indexing sequences of different lengths. *Proceedings of the sixth international conference on Information and knowledge management* (pp. 128–135). ACM.

Bradley, P. S., Fayyad, U., & Reina, C. (1998). Scaling Clustering Algorithms to Large Databases. *Knowledge Discovery and Data Mining*, 9–15.

Brun, M., Sima, C., Hua, J., & Lowey, J. (2007). Model-based evaluation of clustering validation measures. *Pattern Recognition*, *40*(3), 807–824.

Cai, Y., & Ng, R. (2004). Indexing spatio-temporal trajectories with Chebyshev polynomials. *Proceedings of the 2004 ACM SIGMOD international*, 599.

Caiani, E., Porta, A., Baselli, G., Turiel, M., Muzzupappa, S., Pieruzzi, F., Crema, C., et al. (1998). Warped-average template technique to track on a cycle-by-cycle basis the cardiac filling phases on left ventricular volume. *Computers in Cardiology 1998* (pp. 73–76). IEEE.

Carpenter, G. A., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer vision, graphics, and image processing*, *37*(1), 54–115.

Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., Piatetsky-Shapiro, G., et al. (2006). Data mining curriculum. Intensive Working Group of ACM SIGKDD Curriculum Committee, April. Retrieved February 5, 2012, from http://www.acm.org/sigs/sigkdd/curriculum/CURMay06.pdf

Chan, F. K. P., Fu, A. W. C., & Yu, C. (2003). Haar wavelets for efficient similarity search of time-series: with and without time warping. *IEEE Transactions on Knowledge and Data Engineering*, *15*(3), 686–705.

Chan, K., & Fu, A. W. (1999). Efficient time series matching by wavelets. *15th International Conference on Data Engineering, 1999. Proceedings* (Vol. 15, pp. 126–133). IEEE.

Chan, P. K., & Mahoney, M. V. (2005). Modeling multiple time series for anomaly detection. *Fifth IEEE International Conference on Data Mining* (pp. 90–97). IEEE.

Chandrakala, S., & Chandra, C. (2008). A density based method for multivariate time series clustering in kernel feature space. *IEEE International Joint Conference on Neural Networks IEEE World Congress on Computational Intelligence (2008)* (pp. 1885–1890). IEEE.

Chaoji, V., Al Hasan, M., Salem, S., & Zaki, M. J. (2008). Sparcl: Efficient and effective shape-based clustering. *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on* (pp. 93–102). IEEE.

Chen, L., & Ng, R. (2004). On the marriage of lp-norms and edit distance. *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30* (pp. 792–803). VLDB Endowment.

Chen, L., & Özsu, M. T. (2005). Using multi-scale histograms to answer pattern existence and shape match queries. *Time*, *2*(1), 217–226.

Chen, L., Özsu, M. T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data* (pp. 491–502). ACM.

Chen, Q., Chen, L., Lian, X., & Liu, Y. (2007). Indexable PLA for efficient similarity search. *Proceedings of the 33rd international conference on Very large data bases* (pp. 435–446).

Chen, Y., Nascimento, M. A., Ooi, B. C., & Tung, A. K. H. (2007). Spade: On shape-based pattern detection in streaming time series. *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on* (pp. 786–795). IEEE.

Chiş, M., Banerjee, S., & Hassanien, A. E. (2009). Clustering Time Series Data: An Evolutionary Approach. *Foundations of Computational IntelligenceVolume 6*, *6*(1), 193–207.

Chiu, B., Keogh, E., & Lonardi, S. (2003). Probabilistic discovery of time series motifs. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 493–498). ACM.

Chu, S., Keogh, E., Hart, D., Pazzani, M., & others. (2002). Iterative deepening dynamic time warping for time series. *Proc 2nd SIAM International Conference on Data Mining* (pp. 195–212). Citeseer.

Chung, F. L., Fu, T. C., & Luk, R. (2001). Flexible time series pattern matching based on perceptually important points. *Joint Conference on Artificial Intelligence Workshop*, 1–7.

Corduas, M., & Piccolo, D. (2008). Time series clustering and classification by the autoregressive metric. *Computational Statistics & Data Analysis*, *52*(4), 1860–1872.

Corradini, A. (2001). Dynamic time warping for off-line recognition of a small gesture vocabulary. *IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001*. (pp. 82–89).

Dahlhaus, R. (1996). On the Kullback-Leibler information divergence of locally stationary processes. *Stochastic processes and their applications*, *62*(1), 139–168.

Das, G., Gunopulos, D., & Mannila, H. (1997). Finding similar time series. *Principles of Data Mining and Knowledge Discovery*, *1263*(1), 88–100.

Das, G., Lin, K. I., Mannila, H., Renganathan, G., & Smyth, P. (1998). Rule discovery from time series. *Knowledge Discovery and Data Mining*, 16–22.

De Gregorio, A., & Maria Iacus, S. (2010). Clustering of discretely observed diffusion processes. *Computational Statistics & Data Analysis*, *54*(2), 598–606. doi:10.1016/j.csda.2009.10.005

Dembélé, D., & Kastner, P. (2003). Fuzzy C-means method for clustering microarray data. *Bioinformatics*, *19*(8), 973–980.

Ding, C., He, X., Zha, H., & Simon, H. D. (2002). Adaptive dimension reduction for clustering high dimensional data. *International Conference on Data Mining, 2002. ICDM 2002.* (pp. 147–154). IEEE.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., & Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, *1*(2), 1542–1552.

Duan, G., Suzuki, Y., & Kawagoe, K. (2006). Grid Representation of Time Series Data for Similarity Search. *The institute of electronic, Information, and communication Engineer*.

Duan, J., Wang, W., Liu, B., & Xue, Y. (2005). Incorporating with recursive model training in time series clustering. *The Fifth International Conference on Computer and Information Technology CIT05 (2005)*, 105–109.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Cybernetics and Systems*, *3*(3), 32–57.

Ester, M., Kriegel, H. P., & Sander, J. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd International*, *1996*(6), 226–231.

Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *ACM SIGMOD Record*, *23*(2), 419–429.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, *17*(3), 37.

Fayyad, U., Reina, C., & Bradley, P. S. (1998). Initialization of iterative refinement clustering algorithms. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining* (pp. 194–198).

Fern, X. Z., & Brodley, C. E. (2004). Solving cluster ensemble problems by bipartite graph partitioning. *Proceedings of the twenty-first international conference on Machine learning* (p. 36). ACM.

Fink, E., & Pratt, K. B. (2003). *Indexing of compressed time series*. *Data mining in time series databases* (Vol. 57, pp. 43–65). World Scientific Publishing Singapore.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine learning*, *2*(2), 139–172.

Focardi, S. M. M., & others. (2005). *Clustering economic and financial time series: Exploring the existence of stable correlation conditions*. *The Intertek Group* (pp. 1–15). Citeseer.

Fowlkes, E., & Mallows, C. L. (1983). A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, *78*(383), 553–569.

Frentzos, E., Gratsias, K., & Theodoridis, Y. (2007). Index-based most similar trajectory search. *23rd International Conference on Data Engineering, 2007. ICDE 2007. IEEE* (pp. 816–825). IEEE.

Fu, T. C. (2010). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, *24*(1), 164–181.

Fu, T. C., Chung, F. L., Luk, R., & Ng, C. M. (2010). Financial time series indexing based on low resolution clustering. *Proceedings of The 4th IEEE International Conference on Data Mining (ICDM-2004)* (pp. 5–14). Citeseer.

Fu, T. C., Chung, F. L., Ng, V., & Luk, R. (2001). Pattern discovery from stock time series using self-organizing maps. *Workshop Notes of KDD2001 Workshop on Temporal Data Mining* (pp. 26–29). Citeseer.

Fu, T. C., Chung, F., Luk, R., & Ng, C. (2007). Representing financial time series based on data point importance. *Engineering Applications of Artificial Intelligence*, *21*(2), 277–300.

Gan, G., & Ma, C. (2007). *Data clustering: theory, algorithms, and applications*. *ASASIAM Series on Statistics and Applied*.

Gavrilov, M., Anguelov, D., Indyk, P., & Motwani, R. (2000). Mining the stock market : which measure is best? *Proceedings of the sixth ACM SIGKDD*

international conference on Knowledge discovery and data mining (pp. 487–496). ACM.

Ge, X., & Smyth, P. (2000). Deformable Markov model templates for time-series pattern matching. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 81–90). ACM.

Geurts, P. (2001). Pattern extraction for time series classification. *In Proceedings of Principles of Data Mining and Knowledge Discovery, 5th European Conference, Freiburg, Germany, Sept. 3–5* (pp. 115–127).

Ghysels, E., Santa-Clara, P., & Valkanov, R. (2006). Predicting volatility: getting the most out of return data sampled at different frequencies. *Journal of Econometrics*, *131*(1-2), 59–95.

Gionis, A., & Mannila, H. (2003). Finding recurrent sources in sequences. *Proceedings of the seventh annual international conference on Research in computational molecular biology* (pp. 123–130).

Golay, X., Kollias, S., Stoll, G., Meier, D., Valavanis, A., & Boesiger, P. (1998). A new correlation-based fuzzy logic clustering algorithm for FMRI. *Magnetic Resonance in Medicine*, *40*(2), 249–260.

Gordon, A. (1999). Classification. 1999. *Chapman&Hall, CRC, Boca Raton, FL*.

Goutte, C., Toft, P., Rostrup, E., Nielsen, F. Å., Hansen, L. K., & Nielsen. (1999). On Clustering fMRI Time Series* 1. *NeuroImage*, *9*(3), 298–310.

Grass, J. (1996). Anytime algorithm development tools. *ACM SIGART Bulletin*.

Graves D, P. W. (2010). Proximity fuzzy clustering and its application to time series clustering and prediction. *Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications ISDA10* (pp. 49–54).

Gronau, I., & Moran, S. (2007). Optimal implementations of UPGMA and other common clustering algorithms. *Information Processing Letters*, *104*(6).

Guan, H., & Jiang, Q. (2007). Cluster financial time series for portfolio. *International Conference on Wavelet Analysis and Pattern Recognition* (pp. 851 – 856).

Guha, S., Rastogi, R., & Shim, K. (1998). CURE: an efficient clustering algorithm for large databases. *ACM SIGMOD Record* (Vol. 27, pp. 73–84). ACM.

Gullo, F., Ponti, G., Tagarelli, A., & Greco, S. (2009). A time series representation model for accurate and fast similarity detection. *Pattern Recognition*, *42*(11), 2998–3014. doi:10.1016/j.patcog.2009.03.030

Gullo, F., Ponti, G., Tagarelli, A., Tradigo, G., & Veltri, P. (2011). A Time Series Approach for Clustering Mass Spectrometry Data. *Journal of Computational Science*, (2010). doi:10.1016/j.jocs.2011.06.008

Guo, C., Jia, H., & Zhang, N. (2008). Time Series Clustering Based on ICA for Stock Data Analysis. *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, 1–4. doi:10.1109/WiCom.2008.2534

Guo, H., Liu, Y., Liang, H., & Gao, X. (2008). An Application on Time Series Clustering Based on Wavelet Decomposition and Denoising. *2008 Fourth International Conference on Natural Computation*, *1*(4), 419–422. doi:10.1109/ICNC.2008.311

Gupta, L., Molfese, D. L., Tammana, R., & Simos, P. G. (1996). Nonlinear alignment and averaging for estimating the evoked potential. *IEEE Transactions on Biomedical Engineering*, *43*(4), 348–356.

Gusfield, D. (1997). *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press.

Haigh, K., Foslien, W., & Guralnik, V. (2004). Visual Query Language: Finding patterns in and relationships among time series data. *Seventh Workshop on Mining Scientific and Engineering Datasets*, 324–332.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of Intelligent Information*, *17*(2), 107–145.

Han, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Morgan Kaufmann, San Francisco, CA.

Hand, D. J., Mannila, H., & Smyth, P. (2001). *Principles of data mining*. Cambridge, MA: MIT Press.

Harms, S. K., & Goddard, S. (2001). Data mining in a geospatial decision support system for drought risk management. *Proceedings of the 1st national conference on digital government* (pp. 9–16). Los Angeles, CA, 21–23 May: Citeseer.

Harrison, R. (2005). Novel Hybrid Hierarchical-K-means Clustering Method (H-K-means) for Microarray Analysis. *2005 IEEE Computational Systems Bioinformatics Conference - Workshops (CSBW'05)* (pp. 105–108). Ieee.

Hartigan, J. A. (1975). *Clustering Algorithms*. Books on Demand.

Hathaway, R. J., & Bezdek, J. C. (2003). Visual cluster validity for prototype generator clustering models. *Pattern Recognition Letters*, *24*(9-10), 1563–1569.

Hautamaki, V., Nykanen, P., & Franti, P. (2008). Time-series clustering by approximate prototypes. *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on* (pp. 1–4). IEEE.

He, W., Feng, G., Wu, Q., He, T., Wan, S., & Chou, J. (2011). A new method for abrupt dynamic change detection of correlated time series. *International Journal of Climatology*, *32*(10), 1604–1614.

Hirano, S., & Tsumoto, S. (2005). Empirical comparison of clustering methods for long time-series databases. *Active Mining*, *3430*, 268–286.

Hirano, S., & Tsumoto, S. (2007). Cluster analysis of trajectory data on hospital laboratory examinations. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, 324–8.

Honda, R., Wang, S., Kikuchi, T., & Konishi, O. (2002). Mining of moving objects from time-series images and its application to satellite weather imagery. *Journal of Intelligent Information Systems*, *19*(1), 79–93.

Horenko, I. (2010). Finite element approach to clustering of multidimensional time series. *SIAM Journal on Scientific Computing*, *32*(1), 62–83.

Hu, J., Ray, B., & Han, L. (2006). An interweaved hmm/dtw approach to robust time series clustering. *18th International Conference on Pattern Recognition, 2006. ICPR 2006.* (Vol. 3, pp. 145–148). IEEE.

Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. *In Research Issues on Data Mining and Knowledge Discovery*, 1–8.

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, *304*(3), 283–304.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*(1), 193–218.

Huhtala, Y., & Karkkainen, J. (1999). Mining for similarities in aligned time series using wavelets. *Data Mining and Knowledge Discovery: Theory, Tools, and Technology* (pp. 105–393).

Indyk, P., & Koudas, N. (2000). Identifying representative trends in massive time series data sets using sketches. *Proceedings of the 26th International Conference on Very Large Data Bases*, 363–372.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognitionMinimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *23*(1), 67–72.

Ito, F., Hiroyasu, T., Miki, M., & Yokouchi, H. (2009). Detection of preference shift timing using time-series clustering. *2009 IEEE International Conference on Fuzzy Systems* (pp. 1585–1590). Ieee.

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

Jiang, D., & Tang, C. (2004). Cluster analysis for gene expression data: A survey. *Knowledge and Data Engineering,*, *16*(11), 1370–1386.

Kakizawa, Y., Shumway, R. H., & Taniguchi, M. (1998). Discrimination and clustering for multivariate time series. *Journal of the American Statistical Association*, *93*(441), 328–340. doi:10.2307/2669629

Kalpakis, K., Gada, D., & Puttagunta, V. (2001). Distance measures for effective clustering of ARIMA time-series. *Proceedings 2001 IEEE International Conference on Data Mining* (pp. 273–280). IEEE. doi:10.1109/ICDM.2001.989529

Kameda, S., & Yamamura, M. (2006). Spider Algorithm for Clustering Time Series. *World Scientific and Engineering Academy and Society (WSEAS)* (Vol. 2006, pp. 378–383).

Karypis, G., Han, E. H., & Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, *32*(8), 68–75.

Kaufman, L., Rousseeuw, P. J., & Corporation, E. (1990). *Finding groups in data: an introduction to cluster analysis* (Vol. 39). Wiley Online Library.

Kavitha, V., & Punithavalli, M. (2010). Clustering Time Series Data Stream-A Literature Survey. *International Journal of Computer Science and Information Security*, *8*(1), arXiv preprint arXiv:1005.4270.

Kawagoe, K., & Ueda, T. (2002). A similarity search method of time series data with combination of Fourier and wavelet transforms. *Proceedings Ninth International Symposium on Temporal Representation and Reasoning* (pp. 86–92). IEEE Comput. Soc. doi:10.1109/TIME.2002.1027480

Keogh, E. (1997a). A probabilistic approach to fast pattern matching in time series databases. *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining* (pp. 52–57).

Keogh, E. (1997b). Fast similarity search in the presence of longitudinal scaling in time series databases. *International Conference on Tools with Artificial Intelligence* (pp. 578–584).

Keogh, E. (2005). Hot sax: Efficiently finding the most unusual time series subsequence. *Fifth IEEE International Conference on Data Mining ICDM05* (pp. 226–233).

Keogh, E. (2006). A decade of progress in indexing and mining large time series databases. *International Conference on Very Large Data Bases (VLDB)* (pp. 1268–1268).

Keogh, E. (2007). Mining shape and time series databases with symbolic representations. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (p. 1).

Keogh, E., Chakrabarti, K., Pazzani, M., & Mehrotra, S. (2001a). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, *3*(3), 263–286.

Keogh, E., Chakrabarti, K., Pazzani, M., & Mehrotra, S. (2001b). Locally adaptive dimensionality reduction for indexing large time series databases. *ACM SIGMOD Record*, *27*(2), 151–162. doi:10.1145/375663.375680

Keogh, E., Chu, S., & Hart, D. (2004). Segmenting time series: A survey and novel approach. *Data mining in time series databases*, *57*(1), 1–21.

Keogh, E., & Folias, T. (2002). The UCR time series data mining archive. *Riverside CA, http://www. cs. ucr. edu/eamonn/TSDMA*.

Keogh, E., & Kasetty, S. (2003). On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and Knowledge Discovery*, *7*(4), 349–371.

Keogh, E., & Lin, J. (2005). Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and Information Systems*, *8*(2), 154–177. doi:10.1109/ICDM.2003.1250910

Keogh, E., Lonardi, S., & Chiu, B. Y. (2002). Finding surprising patterns in a time series database in linear time and space. *Proceedings of the eighth ACM SIGKDD* (pp. 550–556). ACM.

Keogh, E., Lonardi, S., & Ratanamahatana, C. (2004). Towards parameter-free data mining. *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04* (p. 215). New York, New York, USA: ACM. doi:10.1145/1014052.1014077

Keogh, E., Lonardi, S., Ratanamahatana, C., Wei, L., Lee, S. H., & Handley, J. (2007). Compression-based data mining of sequential data. *Data Mining and Knowledge Discovery*, *14*(1), 99–129.

Keogh, E., & Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Proceedings of the 4th International Conference of Knowledge Discovery and Data Mining* (pp. 239–241). AAAI Press.

Keogh, E., & Pazzani, M. (2000). A simple dimensionality reduction technique for fast similarity search in large time series databases. *LECTURE NOTES IN COMPUTER SCIENCE*, *1805*(1), 122–133.

Keogh, E., & Ratanamahatana, C. (2004). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, *7*(3), 358–386. doi:10.1007/s10115-004-0154-9

Keogh, E., Wei, L., Xi, X., Lee, S. H., & Vlachos, M. (2006). LB_Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures (pp. 882–893). VLDB Endowment.

Kim, S. W., Park, S., & Chu, W. W. (2001). An index-based approach for similarity search supporting time warping in large sequence databases. *Proceedings of International Conference on Data Engineering* (pp. 607–614).

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*(9), 1464–1480.

Korn, F., Jagadish, H. V., & Faloutsos, C. (1997). Efficiently supporting ad hoc queries in large datasets of time sequences. *ACM SIGMOD Record* (Vol. 26, pp. 289–300). ACM.

Košmelj, K., & Batagelj, V. (1990). Cross-sectional approach for clustering time varying data. *Journal of Classification*, *7*(1), 99–109.

Kremer, H., Kranen, P., Jansen, T., Seidl, T., Bifet, A., Holmes, G., & Pfahringer, B. (2011). An effective evaluation measure for clustering on evolving data streams. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 868–876). ACM.

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

Kumar, M., & Patel, N. R. (2002). Clustering seasonality patterns in the presence of errors. *Proceedings of the eighth ACM SIGKDD*, 557–563.

Kumar, N., Lolla, N., Keogh, E., & Lonardi, S. (2005). Time-series bitmaps: a practical visualization tool for working with large time series databases. *SIAM 2005 Data Mining*, 531–535.

Kumar, R., & Nagabhushan, P. (2006). Time Series as a Point-A Novel Approach for Time Series Cluster Visualization. *Conference on Data Mining* (pp. 24–29).

Kurbalija, V., Nachtwei, J., Bernstorff, C. Von, Von Bernstorff, C., Burkhard, H.-D., Ivanović, M., & Fodor, L. (2012). Time-series mining in a psychological domain. *Proceedings of the Fifth Balkan Conference in Informatics* (pp. 58–63). New York, NY, USA: ACM. doi:10.1145/2371316.2371328

Lai, C.-P. P., Chung, P.-C. C., & Tseng, V. S. (2010). A novel two-level clustering method for time series data analysis. *Expert Systems with Applications*, *37*(9), 6319–6326. doi:10.1016/j.eswa.2010.02.089

Lang, W., Morse, M., & Patel, J. M. (2010). Dictionary-based compression for long time-series similarity. *Knowledge and Data Engineering, IEEE Transactions on*, *22*(11), 1609–1622. doi:10.1109/TKDE.2009.201

Larsen, B., & Aone, C. (1999). Fast and effective text mining using linear-time document clustering. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 16–22). ACM.

Last, M., & Kandel, A. (2004). *Data mining in time series databases*. *Genomics and Proteomics*. World scientific.

Latecki, L. J., Megalooikonomou, V., Wang, Q., Lakaemper, R., Ratanamahatana, C., & Keogh, E. (2005). Elastic partial matching of time series. *Knowledge Discovery in Databases: PKDD 2005*, 577–584. doi:10.1109/ICDM.2005.118

Laxman, S., & Sastry, P. S. (2006). A survey of temporal data mining. *Sadhana*, *31*(2), 173–198. doi:10.1007/BF02719780

Leng, M., Lai, X., Tan, G., & Xu, X. (2009). Time series representation for anomaly detection. *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on* (pp. 628–632). IEEE.

Liao, T. W. (2005). Mining of vector time series by clustering.

Liao, T. W., Bolt, B., Forester, J., Hailman, E., Hansen, C., Kaste, R., & O'May, J. (2002). Understanding and projecting the battle state. *23rd Army Science Conference, Orlando, FL* (pp. 2–3).

Liao, T. W., & Ting, C. F. (2006). An adaptive genetic clustering method for exploratory mining of feature vector and time series data. *International Journal of Production Research*, *44*(14), 2731–2748.

Lin, J., Etter, D., & DeBarr, D. (2008). Exact and approximate reverse nearest neighbor search for multimedia data. *International Conference on Data Mining* (pp. 656–667).

Lin, J., Keogh, E., Lonardi, S., & Chiu, B. (2003). A symbolic representation of time series, with implications for streaming algorithms. *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery - DMKD '03*, 2. doi:10.1145/882085.882086

Lin, J., Keogh, E., Lonardi, S., Lankford, J., & Nystrom, D. (2004). Visually mining and monitoring massive time series. *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, 460. doi:10.1145/1014052.1014104

Lin, J., Keogh, E., & Truppel, W. (2003). Clustering of streaming time series is meaningless. *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery DMKD 03*, 56. doi:10.1145/882082.882096

Lin, J., Keogh, E., Wei, L., & Lonardi, S. (2007). Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, *15*(2), 107–144. doi:10.1007/s10618-007-0064-z

Lin, J., & Li, Y. (2009). Finding structural similarity in time series data using bag-of-patterns representation. *Proceedings of the 21st International Conference on Scientific and Statistical Database Management* (pp. 461–477). Springer-Verlag.

Lin, J., Vlachos, M., Keogh, E., & Gunopulos, D. (2004). Iterative incremental clustering of time series. *Advances in Database Technology-EDBT 2004*, 521–522.

Lin, J., Vlachos, M., Keogh, E., Gunopulos, D., Liu, J., Yu, S., & Le, J. (2005). A MPAA-based iterative clustering algorithm augmented by nearest neighbors search for time-series data streams. *Advances in Knowledge Discovery and Data Mining*, 369–377.

Lin, R., & Shim, H. S. S. K. (1995). Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *Proc. of the 21st VLDB Conference*, 490–501.

Lin, S., Song, M., & Zhang, L. (2008). Comparison of cluster representations from partial second-to full fourth-order cross moments for data stream clustering. *Eighth IEEE International Conference on Data Mining, 2008. ICDM '08* (pp. 560–569).

Liu, W., & Shao, L. (2009). Research of SAX in Distance Measuring for Financial Time Series Data. *First International Conference on Information Science and Engineering* (pp. 935–937). Ieee. doi:10.1109/ICISE.2009.924

Lkhagva, B., Suzuki, Y. u., & Kawagoe, K. (2006). New time series data representation ESAX for financial applications. *Proceedings. 22nd International Conference onData Engineering Workshops, 2006.* (p. x115). IEEE. doi:10.1109/ICDEW.2006.99

Lonardi, S. (2001). *Global detectors of unusual words: design, implementation, and applications to pattern discovery in biosequences*. Purdue University.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium Mathematical Statist. Probability* (Vol. 1, pp. 281–297).

Manganaro, V., Paratore, S., Alessi, E., Coffa, S., & Cavallaro, S. (2005). Adding semantics to gene expression profiles: new tools for drug discovery. *Current medicinal chemistry*, *12*(10), 1149–1160.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to information retrieval*. *Online* (Vol. 1). Cambridge University Press Cambridge.

Meila, M. (2003). Comparing clusterings by the variation of information. In B. Schölkopf & M. Warmuth (Eds.), *Comparing Clusterings by the Variation of Information Learning Theory and Kernel Machines* (pp. 173–187). Springer.

Milligan, G. (1981). A Monte Carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*, *46*(2), 187–199.

Milligan, G. W., & Cooper, M. C. (1986). A study of the comparability of external criteria for hierarchical cluster analysis. *A study of the comparability of external criteria for hierarchical cluster analysis*, *21*(4), 441–458.

Minnen, D., Isbell, C. L., Essa, I., & Starner, T. (2007). Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, *22*(1), 615.

Minnen, D., Starner, T., Essa, M., & Isbell, C. (2006). Discovering characteristic actions from on-body sensor data. *10th IEEE International Symposium on Wearable Computers* (pp. 11–18).

Mirkin, B. (2005). *Clustering for A Data Recovery Approach*. *New York*. Chapman & Hall/CRC.

Mitsa, T. (2009). *Temporal data mining*. *Drug safety : an international journal of medical toxicology and drug experience* (Vol. 33). Chapman & Hall/CRC. doi:10.2165/11537630-000000000-00000

Möller-Levet, C., Klawonn, F., Cho, K. H., & Wolkenhauer, O. (2003). Fuzzy clustering of short time-series and unevenly distributed sampling points. *Advances in Intelligent Data Analysis V*, 330–340.

Mörchen, F., & Ultsch, A. (2005). Optimizing time series discretization for knowledge discovery. *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining - KDD '05*, 660. doi:10.1145/1081870.1081953

Morchen, F., Ultsch, A., Mörchen, F., & Hoos, O. (2005). Extracting interpretable muscle activation patterns with time series knowledge mining. *JOURNAL OF KNOWLEDGE BASED*, *9*(3), 197–208.

Morinaka, Y., Yoshikawa, M., Amagasa, T., & Uemura, S. (2001). The L-index: An indexing structure for efficient subsequence matching in time sequence databases. *Proc. 5th PacificAisa Conf. on Knowledge Discovery and Data Mining* (pp. 51–60). Citeseer.

Morris, B., & Trivedi, M. (2009). Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 312–319. doi:10.1109/CVPR.2009.5206559

Morse, M. D., & Patel, J. M. (2007). An efficient and accurate method for evaluating time series similarity. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data SIGMOD 07* (p. 569). ACM Press. doi:10.1145/1247480.1247544

Murtagh, F. (1985). Multidimensional clustering algorithms. *Compstat Lectures, Vienna: Physika Verlag, 1985*, *1*.

Nakhaeizadeh, G., Steurer, E., & Bartlmae, K. (2002). BANKING AND FINANCE. *Handbook of data mining and knowledge discovery* (pp. 771–780).

Ng, R. T., & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. *Proceedings of the International Conference on Very Large Data Bases* (pp. 144–144). INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE).

Niennattrakul, V., & Ratanamahatana, C. (2007a). On clustering multimedia time series data using k-means and dynamic time warping. *International Conference on Multimedia and Ubiquitous Engineering, 2007. MUE '07.* (pp. 733–738). IEEE Computer Society.

Niennattrakul, V., & Ratanamahatana, C. (2007b). Inaccuracies of shape averaging method using dynamic time warping for time series data. *Computational Science–ICCS 2007*, 513–520.

Oates, T. (1999). Identifying distinctive subsequences in multivariate time series by clustering. *Proceedings of the fifth ACM SIGKDD international* (pp. 322–326).

Oates, T., Firoiu, L., & Cohen, P. (2001). Using dynamic time warping to bootstrap HMM-based clustering of time series. *SEQUENCE LEARNING: PARADIGMS, ALGORITHMS, AND APPLICATIONS*, *1*(1828), 35–52.

Oates, T., Schmill, M. D., & Cohen, P. R. (2000). A method for clustering the experiences of a mobile robot that accords with human judgments. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE* (pp. 846–851). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Owsley, L. M. D., Atlas, L. E., & Bernard, G. D. (1997). Self-organizing feature maps and hidden Markov models for machine-tool monitoring. *IEEE Transactions on Signal Processing*, *45*(11), 2787–2798.

Panuccio, A., Bicego, M., & Murino, V. (2002). A Hidden Markov Model-based approach to sequential data clustering. In T. Caelli, A. Amin, R. Duin, R. De, & M. Kamel (Eds.), *Structural, Syntactic, and Statistical Pattern Recognition*.

Park, S., & Lee, J. (2010). Representation and clustering of time series by means of segmentation based on PIPs detection. *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, 17–21. doi:10.1109/ICCAE.2010.5451841

Pavlidis, N., Plagianakos, V. P., Tasoulis, D. K., & Vrahatis, M. N. (2006). Financial forecasting through unsupervised clustering and neural networks. *Operational Research*, *6*(2), 103–127.

Perng, C. S., Wang, H., Zhang, S. R., & Parker, D. S. (2000). Landmarks: a new model for similarity-based pattern querying in time series databases. *Data Engineering, 2000. Proceedings. 16th International Conference on* (pp. 33–42). IEEE.

Petitjean, F., Ketterlin, A., & Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, *44*(3), 678–693. doi:10.1016/j.patcog.2010.09.013

Phetking, C., Sap, M. N. M., & Selamat, A. (2008). A multiresolution important point retrieval method for financial time series representation. *International Conference on Computer and Communication Engineering* (pp. 510–515). IEEE. doi:10.1109/ICCCE.2008.4580656

Policker, S., & Geva, A. B. B. (2000). Nonstationary time series analysis by temporal clustering. *Systems, Man, and Cybernetics, Part B:*, *30*(2), 339–343.

Polz, P. M., Hortnagl, E., & Prem, E. (2003). *Processing and Clustering Time Series of Mobile Robot Sensory Data. Technical Report, Österreichisches Forschungsinstitut für Artificial Intelligence, Wien, TR-2003-10, 2003. Time.*

Popivanov, I., & Miller, R. J. (2002). Similarity search over time-series data using wavelets. *ICDE '02: Proceedings of the 18th International Conference on Data Engineering* (pp. 212–224). Citeseer.

Portet, F., Reiter, E., Gatt, A., Hunter, J., Sripada, S., Freer, Y., & Sykes, C. (2009). Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, *173*(7), 789–816.

Pratt, K. B., & Fink, E. (2002). Search for patterns in compressed time series. *International Journal of Image and Graphics*, *2*(1), 89–106.

Qian, J., Dolled-Filhart, M., Lin, J., Yu, H., & Gerstein, M. (2001). Beyond synexpression relationships: local clustering of time-shifted and inverted gene expression profiles identifies new, biologically relevant interactions1. *Journal of Molecular Biology*, *314*(5), 1053–1066. doi:10.1006/jmbi.2001.5219

Rabiner, L., & Levinson, S. (1979). Speaker-independent recognition of isolated words using clustering techniques. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *27*(4), 336–349.

Rai, P., & Singh, S. (2010). A Survey of Clustering Techniques. *International Journal of Computer Applications IJCA*, *7*(12), 1–5.

Rakthanmanon, T., Campana, A. B., Batista, G., Zakaria, J., & Keogh, E. (2012). Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping. *Conference on Knowledge Discovery and Data Mining* (pp. 262–270).

Ramoni, M., Sebastiani, P., & Cohen, P. (2000). Multivariate clustering by dynamics. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE* (pp. 633–638). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, *66*(336), 846–850.

Rani, S., & Sikka, G. (2012). Recent Techniques of Clustering of Time Series Data: A Survey. *International Journal of Computer Applications*, *52*(15), 1–9. doi:10.5120/8282-1278

Ratanamahatana, C. (2005). Multimedia retrieval using time series representation and relevance feedback. *Proceedings of 8th International Conference on Asian Digital Libraries (ICADL2005)* (pp. 400–405).

Ratanamahatana, C., & Keogh, E. (2004a). Everything you know about dynamic time warping is wrong. *Review Literature And Arts Of The Americas*, *35*(9), S535–40.

Ratanamahatana, C., & Keogh, E. (2004b). Making time-series classification more accurate using learned constraints. *Proceedings of SIAM International Conference on Data Mining* (pp. 11–22). Lake Buena Vista, Florida.

Ratanamahatana, C., & Keogh, E. (2005). Three myths about dynamic time warping data mining. *International Conference on Data Mining (SDM'05)* (pp. 506–510).

Ratanamahatana, C., Keogh, E., Bagnall, A. J., & Lonardi, S. (2005). A Novel Bit Level Time Series Representation with Implications for Similarity Search and Clustering. *In Proc. 9th Pacific-Asian Int. Conf. on Knowledge Discovery and Data Mining (PAKDD'05}* (pp. 771–777). Springer.

Ratanamahatana, C., & Niennattrakul, V. (2006). Clustering Multimedia Data Using Time Series. *International Conference on Hybrid Information Technology, 2006. ICHIT '06.* (pp. 372–379).

Rauber, A., Pampalk, E., & Paralič, J. (2000). Empirical evaluation of clustering algorithms. *Journal of Information and Organizational Sciences*, *24*(2), 195–209.

Rebbapragada, U., Protopapas, P., Brodley, C. E., & Alcock, C. (2009). Finding anomalous periodic time series. *Machine learning*, *74*(3), 281–313.

Reinert, G., Schbath, S., & Waterman, M. S. (2000). Probabilistic and statistical properties of words: an overview. *Journal of Computational Biology*, *7*(1-2), 1–46.

Rodgers, J. L., & Nicewander, W. A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, *42*(1), 59–66.

Rohlf, F. (1974). Methods of comparing classifications. *Annual Review of Ecology and Systematics*, 101–113.

Rosenberg, A., & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* (pp. 410–420).

Saito, N. (2000). Local feature extraction and its applications using a library of bases. *Topics in Analysis and Its Applications: Selected Theses*, 269–451.

Sakoe, H., & Chiba, S. (1971). A dynamic programming approach to continuous speech recognition. *Proceedings of the Seventh International Congress on Acoustics* (Vol. 3, pp. 65–69).

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *26*(1), 43–49.

Sakurai, Y., Yoshikawa, M., & Faloutsos, C. (2005). FTW: fast similarity search under the time warping distance. *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (Vol. 1, pp. 326–337). ACM.

Salvador, S., & Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, *11*(5), 561–580.

Sankoff, D., & Kruskal, J. B. (1983). Time warps, string edits, and macromolecules: the theory and practice of sequence comparison. *Reading: Addison-Wesley Publication, 1983, edited by Sankoff, David; Kruskal, Joseph B.*, *1*.

Seo, J., & Shneiderman, B. (2002). Interactively exploring hierarchical clustering results. *Computer*, *35*(7), 80–86.

Sfetsos, A., & Siriopoulos, C. (2004). Time series forecasting with a hybrid clustering scheme and pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics, 34*(3), 399–405.

Shahabi, C., Tian, X., & Zhao, W. (2002). Tsa-tree: A wavelet-based approach to improve the efficiency of multi-level surprise and trend queries on time-series data (pp. 55–68). IEEE.

Shatkay, H., & Zdonik, S. B. (1996). Approximate queries and representations for large data sequences. *Data Engineering, 1996. Proceedings of the Twelfth International Conference on* (pp. 536–545). IEEE.

Shavlik, J. W., & Dietterich, T. G. (1990). *Readings in machine learning*. Morgan Kaufmann.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). Wavecluster: A multi-resolution clustering approach for very large spatial databases. *PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES* (pp. 428–439).

Shi, J. (2000). Normalized cuts and image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *22*(8), 888–905.

Shieh, J., & Keogh, E. (2009). iSAX: disk-aware mining and indexing of massive time series datasets. *Data Mining and Knowledge Discovery*, *19*(1), 24–57. doi:10.1007/s10618-009-0125-6

Shumway, R. H. R. (2003). Time-frequency clustering and discriminant analysis. *Statistics & probability letters*, *63*(3), 307–314. doi:10.1016/S0167-7152(03)00095-6

Smyth, P. (1997). Clustering sequences with hidden Markov models. *Advances in neural information processing systems*, (9), 648–654.

Sneath, P., & Sokal, R. (1973). *Numerical taxonomy. The principles and practice of numerical classification.* San Francisco, CA.: W. H. Freeman.

Steinbach, M., Tan, P. N., Kumar, V., Klooster, S., & Potter, C. (2003). Discovery of climate indices using clustering. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 446–455).

Steinley, D. (2004). Properties of the Hubert-Arable Adjusted Rand Index. *Psychological Methods*, *9*(3), 386.

Strehl, A., & Ghosh, J. (2003). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, *3*, 583–617.

Studholme, C., Hill, D. L. G., & Hawkes, D. J. (1999). An overlap invariant entropy measure of 3D medical image alignment. *Pattern recognition*, *32*(1), 71–86.

Tan, P.-N., Steinbach, M., & Kumar, V. (2006). *Cluster analysis: basic concepts and algorithms*. *Introduction to data mining* (pp. 487–567).

Tran, D., & Wagner, M. (2002). Fuzzy c-means clustering-based speaker verification. *Advances in Soft Computing—AFSS 2002*, (2275), 363–369.

Tseng, V. S., & Kao, C. P. (2005). Efficiently mining gene expression data via a novel parameterless clustering method. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *2*(4), 355–365.

Tseng, V. S., & Kao, C.-P. P. (2007). A Novel Similarity-Based Fuzzy Clustering Algorithm by Integrating PCM and Mountain Method. *IEEE Transactions on Fuzzy Systems*, *15*(6), 1188–1196. doi:10.1109/TFUZZ.2006.890673

Uehara, K., & Shimada, M. (2002). Extraction of primitive motion and discovery of association rules from human motion data. *Progress in Discovery Science*, 73–88.

Ultsch, A., & Mörchen, F. (2005). *ESOM-Maps: tools for clustering, visualization, and classification with Emergent SOM*.

Van Rijsbergen, C. J. (1979). *Information retrieval*. Butterworths, London Boston.

Van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The computer journal*, *29*(6), 481–485.

Van Wijk, J. J., & Van Selow, E. R. (1999). Cluster and calendar based visualization of time series data. *Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis'99)*, 4–9,. doi:10.1109/INFVIS.1999.801851

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, *11*(1), 37–57.

Vlachos, M., Gunopulos, D., & Das, G. (2004). Indexing time-series under conditions of noise. In M. Last, A. Kandel, & H. Bunke (Eds.), *Data Mining in Time Series Databases*, *World Scientific*, *Singapore* (p. 67).

Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. *Data Engineering, 2002. Proceedings. 18th International Conference on* (pp. 673–684). IEEE. doi:10.1109/ICDE.2002.994784

Vlachos, M., Lin, J., & Keogh, E. (2003). A wavelet-based anytime algorithm for k-means clustering of time series. *Proc. Workshop on Clustering*, 23–30.

Vuori, V., & Laaksonen, J. (2002). A comparison of techniques for automatic clustering of handwritten characters. *Pattern Recognition, 2002., 3*, 30168.

Wang, C., & Sean Wang, X. (2000). Supporting content-based searches on time series via approximation. *Scientific and Statistical Database* (pp. 69–81).

Wang, H., Wang, W., Yang, J., & Yu, P. S. (2002). Clustering by pattern similarity in large data sets. *Proceedings of the 2002 ACM SIGMOD international conference on Management of data* (pp. 394–405). ACM.

Wang, W., Yang, J., & Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining. *Proceedings of the International Conference on Very Large Data Bases* (pp. 186–195). INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE).

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

Wang, X., Smith, K. A., & Hyndman, R. J. (2005). Dimension reduction for clustering time series using global characteristics. *Computational Science–ICCS 2005*, 792–795.

Wang, X., Smith, K., & Hyndman, R. (2006). Characteristic-Based Clustering for Time Series Data. *Data Mining and Knowledge Discovery*, *13*(3), 335–364. doi:10.1007/s10618-005-0039-x

Wang, Xiaoyue, Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., & Keogh, E. (2012). Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, Springer Netherlands. doi:10.1007/s10618-012-0250-5

Wang, Z. J., & Willett, P. (2004). Joint segmentation and classification of time series using class-specific features. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *34*(2), 1056–1067.

Warrenliao, T. (2005). Clustering of time series data--a survey. *Pattern Recognition*, *38*(11), 1857–1874. doi:10.1016/j.patcog.2005.01.025

Wei, L., Kumar, N., Lolla, V., & Keogh, E. (2005). Assumption-free anomaly detection in time series. *Proceedings of the 17th International Conference on Scientific and Statistical Database Management* (pp. 237–240).

Wismüller, A., Lange, O., Dersch, D. R., Leinsinger, G. L., Hahn, K., Pütz, B., & Auer, D. (2002). Cluster analysis of biomedical image time-series. *International Journal of Computer Vision*, *46*(2), 103–128.

Wu, J., Xiong, H., & Chen, J. (2009). Adapting the right measures for k-means clustering. *Proceedings of the 15th ACM SIGKDD* (pp. 877–886).

Wu, Y. L., Agrawal, D., & El Abbadi, A. (2000). A comparison of DFT and DWT based similarity search in time-series databases (pp. 488–495). ACM.

Xi, X., Keogh, E., Shelton, C., Wei, L., & Ratanamahatana, C. (2006). Fast time series classification using numerosity reduction. *Proceedings of the 23rd international conference on Machine learning* (pp. 1033–1040). ACM.

Xiong, Y., & Yeung, D. Y. (2002). Mixtures of ARMA models for model-based time series clustering. *Data Mining, 2002. ICDM 2003.* (pp. 717–720).

Xiong, Y., & Yeung, D. Y. (2004). Time series clustering with ARMA mixtures. *Pattern Recognition*, *37*(8), 1675–1689. doi:10.1016/j.patcog.2003.12.018

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, *16*(3), 645–78. doi:10.1109/TNN.2005.845141

Yeung, K., Fraley, C., Murua, A., Raftery, A. E., & Ruzzo, W. L. (2001). Model-based clustering and data transformations for gene expression data. *Bioinformatics*, *17*(10), 977–987.

Yeung, K., Haynor, D., & Ruzzo, W. (2001). Validating clustering for gene expression data. *Bioinformatics*, *17*(4), 309–318.

Yi, B. K., & Faloutsos, C. (2000). Fast time sequence indexing for arbitrary Lp norms. *Proceedings of the 26th international conference on* (pp. 385–394). VLDB.

Yi, B. K., Jagadish, H. V., & Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time warping. *Data Engineering, 1998. Proceedings., 14th International Conference on* (pp. 201–208). IEEE.

Yin, J., & Yang, Q. (2005). Integrating hidden Markov models and spectral analysis for sensory time series clustering. *Fifth IEEE International Conference on Mining* (pp. 8–18). IEEE. doi:10.1109/ICDM.2005.82

Yu, F., Dong, K., Chen, F., Jiang, Y., & Zeng, W. (2007). Clustering Time Series with Granular Dynamic Time Warping Method. *2007 IEEE International Conference on Granular Computing GRC 2007*, 393–393. doi:10.1109/GrC.2007.34

Zakaria, J., Rotschafer, S., Mueen, A., Razak, K., & Keogh, E. (2012). Mining Massive Archives of Mice Sounds with Symbolized Representations. *SIGKDD* (pp. 1–10).

Zhang, H., Ho, T. B., Zhang, Y., & Lin, M. S. (2006). Unsupervised Feature Extraction for Time Series Clustering Using Orthogonal Wavelet Transform. *INFORMATICA*, *30*(3), 305–319.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD Record* (Vol. 25, pp. 103–114). ACM.

Zhang, X., Liu, J., Du, Y., & Lv, T. (2011). A novel clustering method on time series data. *Expert Systems with Applications*, *38*(9), 11891–11900. doi:10.1016/j.eswa.2011.03.081

Zhang, X., Wu, J., Yang, X., Ou, H., & Lv, T. (2009). A novel pattern extraction method for time series classification. *Optimization and Engineering*, *10*(2), 253–271.

Zhang, Z., Huang, K., & Tan, T. (2006). Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. *18th International Conference on Pattern Recognition, ICPR 2006*, *3*, 1135–1138.

Zhao, Y., & Karypis, G. (2004). Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, *55*(3), 311–331.

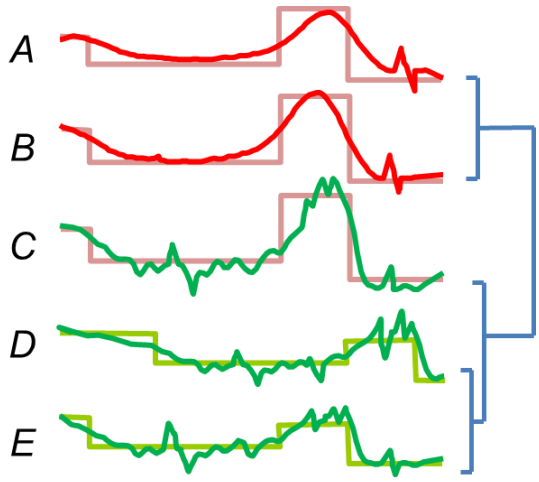
Zhu, Q., Rakthanmanon, Q., Batista, G. E. A. P. A., & Keogh, E. J. (2012). A Novel Approximation to Dynamic Time Warping allows Anytime Clustering of Massive Time Series Datasets. *SIAM International Conference on Data Mining* (pp. 999–1010).

Zhu, Y., & Shasha, D. (2003). Warping indexes with envelope transforms for query by humming. *In Proceedings of the 2003 ACM SIGMOD international conference on Management of data* (pp. 181–192). ACM.

Zilberstein, S., & Russell, S. (1995). Approximate reasoning using anytime algorithms. In S. Natarajan (Ed.), *The Springer International Series in Engineering and Computer Science* (pp. 43–62). Springer US.

APPENDIX A

In banking systems, for example, the customers who commits banking fraud, have usually *frequent transactions* in a short period. This concept helps experts in fraud detection to find potential criminals in banking systems. *Frequent transactions* can be considered from two points of view; first, *frequent transactions* (of a customer) which are more frequent rather than his previous or later transactions. It is same as abnormal observation in a time-series which can be detected by monitoring or anomaly detection process. Second, *frequent transactions* which indicate rare behavior (of a customer) in comparison with similar customers in the same cluster. It is usually detected by clustering of customer. However, because *frequent transactions* are committed in a short period in comparison with whole their time-series, it is usually overlooked in approximation process. The second case is focused here, i.e., the case that behavior of a customer differs from other members of the same cluster. A solution to address this kind of problem, is grouping the users in more accurate clusters, then, the rare cases can be detected as outliers time-series (or new clusters). For example, following figure shows five customers by their transactions. Considering approximated time-series (brown lines), it seems that customer C belong to cluster (A, B). However, customer c is mostly belongs to cluster (E and D) in terms of the frequency. In a sensitive system, such as banking system, it should be detected as an outlier or a sub cluster of main groups (to be decided by experts) because it contains a behaviour which is rare rather than its homogenous customer in the same cluster. These sorts of accurate clusters cannot achieve by conventional clustering of approximated time-series obtained by dimensionally reduction methods.

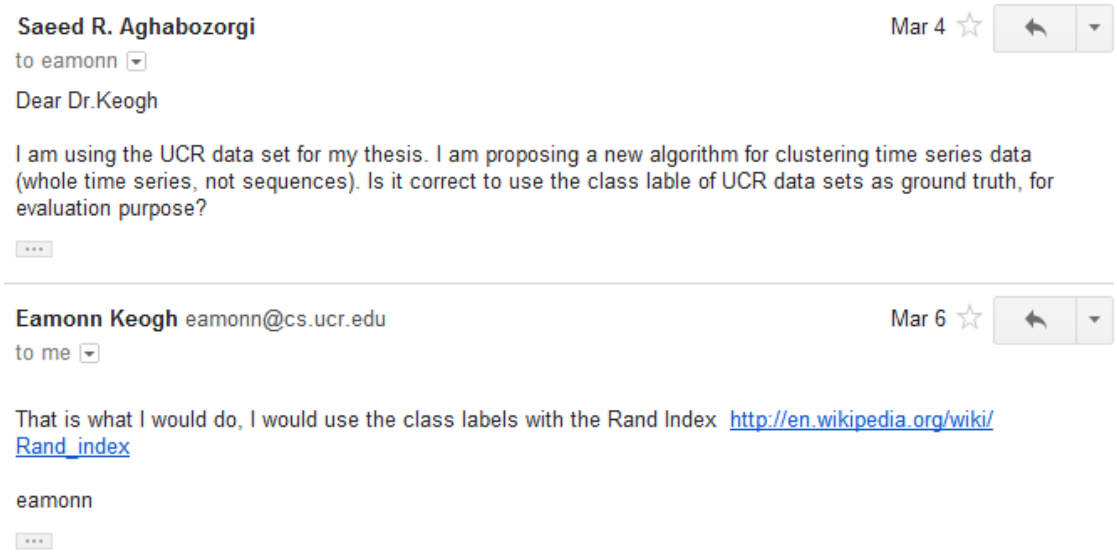


Detecting different levels of dissimilarity in a cluster

Another example is daily stock where if one consider the daily stock instead hourly stock time-series, some important times might be overlooked such as extremes or peaks which it reflects the quality of clusters .

APPENDIX B

On evaluation of clustering of UCR dataset:



Saeed R. Aghabozorgi Mar 4

to eamonn

Dear Dr.Keogh

I am using the UCR data set for my thesis. I am proposing a new algorithm for clustering time series data (whole time series, not sequences). Is it correct to use the class lable of UCR data sets as ground truth, for evaluation purpose?

...

Eamonn Keogh eamonn@cs.ucr.edu Mar 6

to me

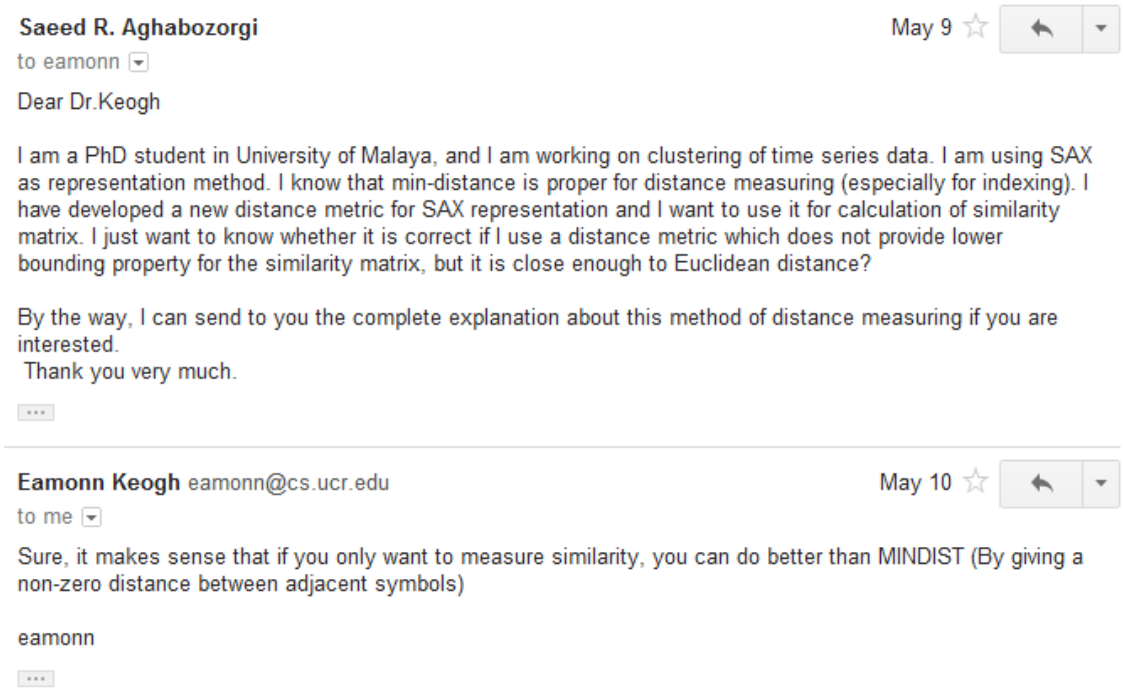
That is what I would do, I would use the class labels with the Rand Index http://en.wikipedia.org/wiki/Rand_index

eamonn

...

APPENDIX C

Comment about using a new distance method (instead of MINDIST) for clustering purpose in the case that representation method is SAX.



Saeed R. Aghabozorgi May 9

to eamonn

Dear Dr.Keogh

I am a PhD student in University of Malaya, and I am working on clustering of time series data. I am using SAX as representation method. I know that min-distance is proper for distance measuring (especially for indexing). I have developed a new distance metric for SAX representation and I want to use it for calculation of similarity matrix. I just want to know whether it is correct if I use a distance metric which does not provide lower bounding property for the similarity matrix, but it is close enough to Euclidean distance?

By the way, I can send to you the complete explanation about this method of distance measuring if you are interested.
Thank you very much.

Eamonn Keogh eamonn@cs.ucr.edu May 10

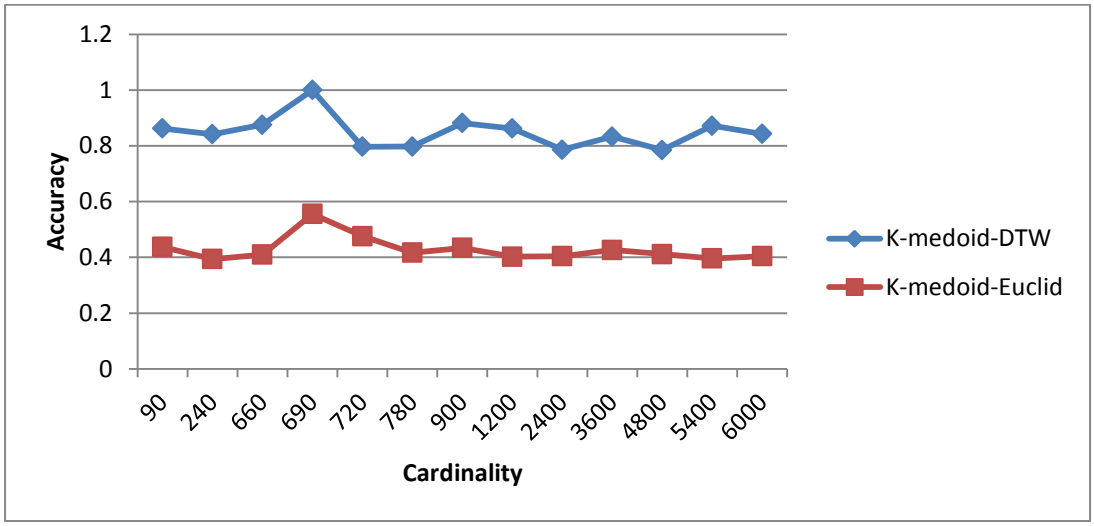
to me

Sure, it makes sense that if you only want to measure similarity, you can do better than MINDIST (By giving a non-zero distance between adjacent symbols)

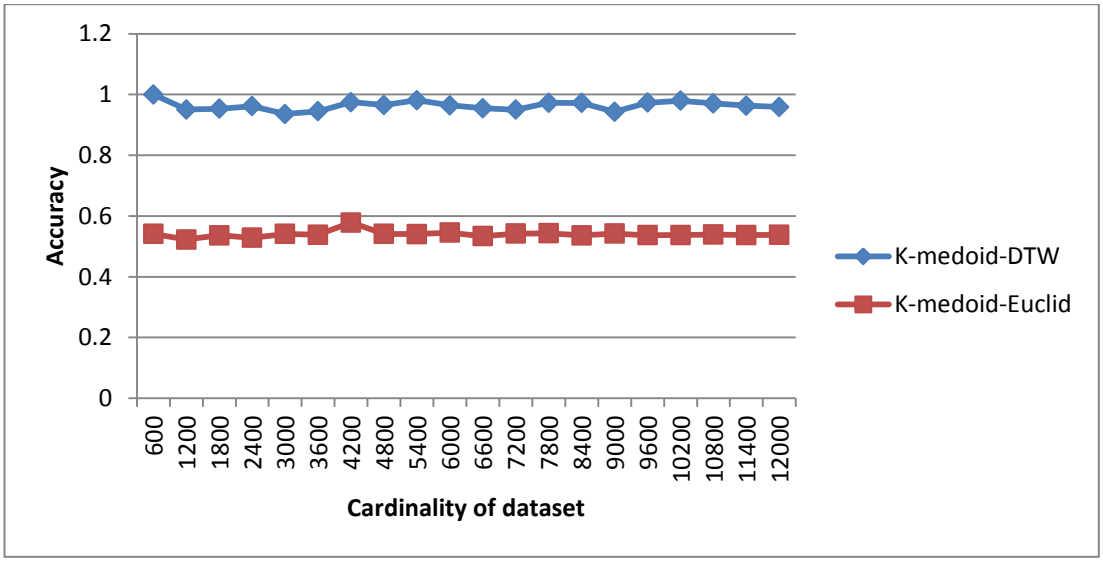
eamonn

APPENDIX D

Difference between two most-used distance measures in quality of time-series clustering is shown in the following plots. The results are related to DTW and ED used in k-Medoids clustering of CBF and CC dataset with different cardinalities.



The quality of clustering of CBF using ED and DTW

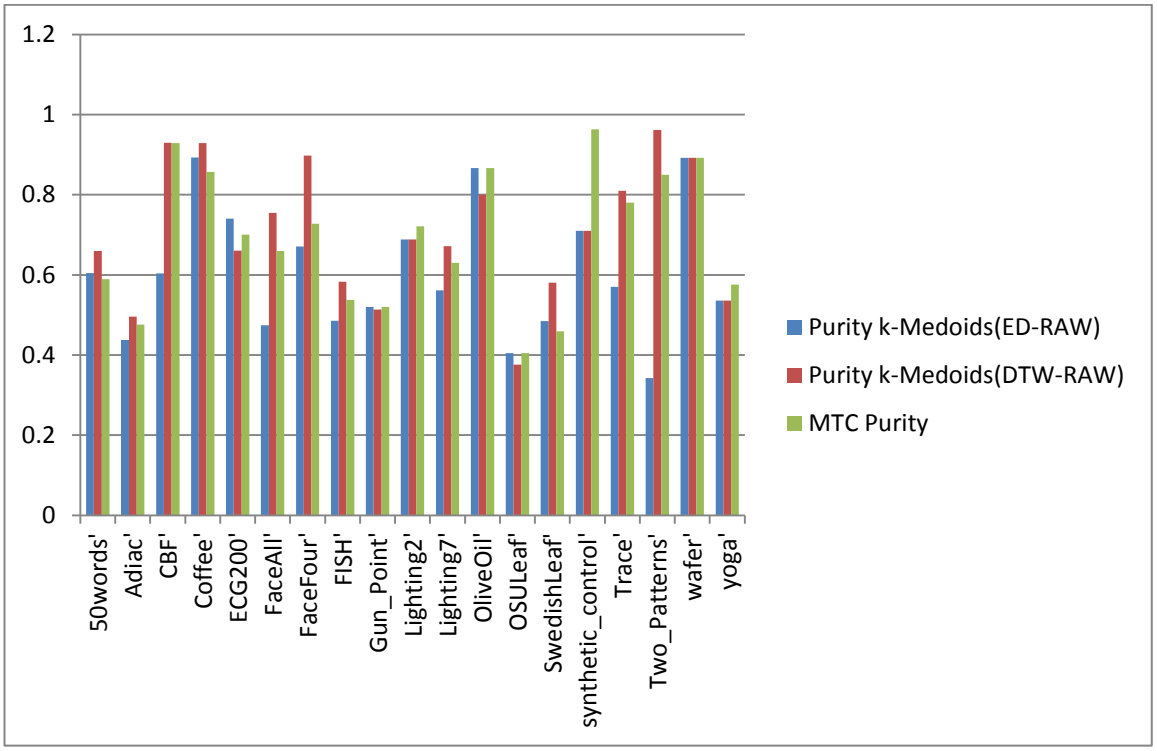


The quality of clustering of CC using ED and DTW

The charts indicate two things: first, quality of DTW is higher than ED for all cardinalities. Secondly, in contrast to indexing and classification problem, quality of clustering using DTW and ED is quite different even for large datasets.

APPENDIX E

To calculate distance between time-series data, ED and DTW are used as distance metric. In the following chart the maximum purity of MTC approach in front of maximum purity of k-Medoids approach (using raw time-series) is shown. It is the result of 100 run on the same datasets. The purity measure is used to compare with rival works such as (Ratanamahatana & Niennattrakul, 2006). However, the dataset used by Ratanamahatana and Niennattrakul is different from the existing and published dataset in UCR repository. As a result, the purity of the datasets is calculated and reported using the published datasets.

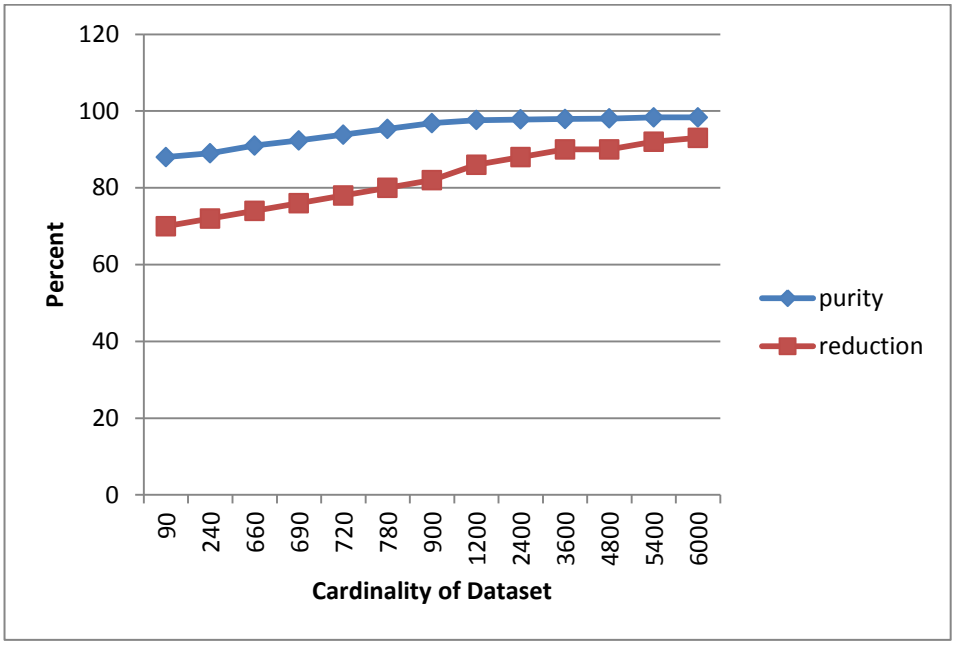


In (H. Zhang et al., 2006), the mean of the evaluation criteria values obtained from 100 runs of k-Means algorithm with the extracted features are reported. The results are compared with MTC in the following table.

	Mean RI		Mean jaccard		Mean FM		Mean CSM		Mean NMI	
	(H. Zhang et al., 2006)	MTC	(H. Zhang et al., 2006)	MTC	(H. Zhang et al., 2006)	MTC	(H. Zhang et al., 2006)	MTC	(H. Zhang et al., 2006)	MTC
CBF	0.6447	0.844	0.3439	0.6388	0.5138	0.7769	0.5751	0.8577	0.3459	0.676
CC	0.8514	0.9311	0.4428	0.6918	0.6203	0.8136	0.6681	0.8382	0.6952	0.8479
Trace	0.7498	0.8112	0.3672	0.4856	0.5400	0.656	0.5537	0.6648	0.5187	0.6297
Gun	0.4975	0.4972	0.3289	0.3291	0.4949	0.4953	0.5000	0.5146	0.0000	0.0007
ECG	0.4919	0.5436	0.2644	0.4722	0.4314	0.6498	0.4526	0.5454	0.0547	0.032

APPENDIX F

PCS provides a good trade-off between quality and reduction-rate dynamically. Here, the experiment is performed on CBF with different cardinalities to show the effect of PCS on large time-series. As expected, the PCS works very well on large datasets.



Proportion of purity and reduction rate on different cardinalities of CBF

APPENDIX G

In the experiment results reported in this section, the range of values (shown in Table 5.2) is used as the parameter values of MTC in applying on all datasets. As mentioned, for different parameter combinations, the maximum quality is reported in following table.

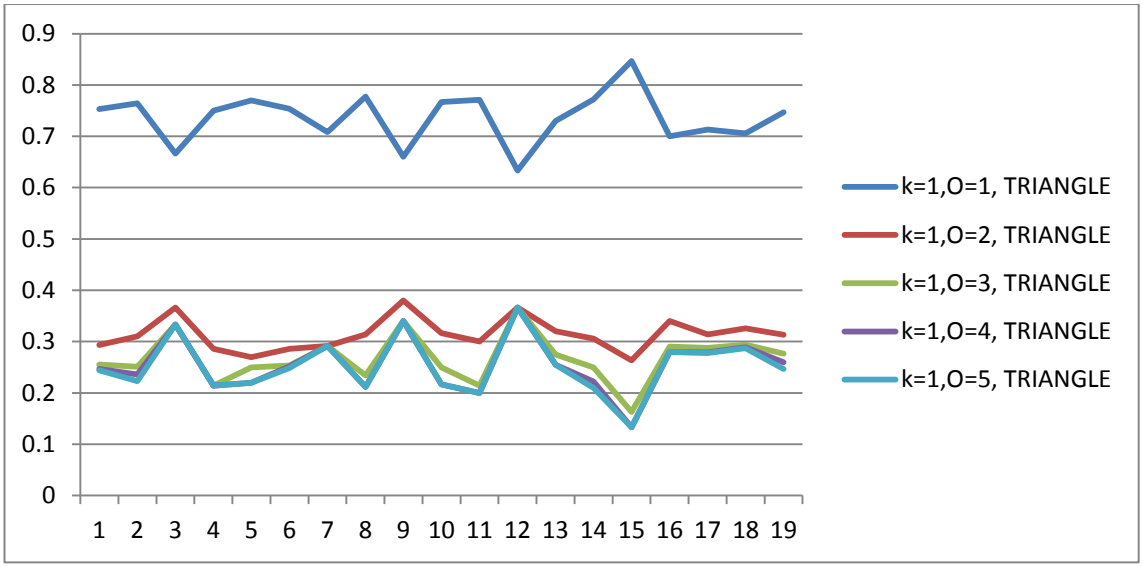
The result of MTC in comparison with conventional algorithms (maximum accuracy)

Algorithm / Dataset	k-Medoids	MTC k-Medoids	Hier (Avg)	MTC Hier-Avg	Hier (Single)	MTC Hier-Single
50words	**0.517**	**0.617**	**0.514**	**0.583**	**0.297**	**0.349**
Adiac	**0.363**	**0.531**	**0.289**	**0.52**	**0.294**	**0.292**
CBF	**0.484**	**0.841**	**0.39**	**0.673**	0.347	0.344
Coffee	**0.388**	**0.655**	**0.398**	**0.587**	**0.392**	**0.43**
ECG200	0.512	0.457	0.544	0.448	**0.476**	**0.476**
FaceAll	**0.407**	**0.631**	**0.335**	**0.559**	**0.199**	**0.229**
FaceFour	**0.573**	**0.753**	0.642	0.541	0.428	0.404
FISH	**0.397**	**0.492**	**0.313**	**0.468**	**0.234**	**0.236**
Gun_Point	**0.324**	**0.367**	**0.431**	**0.42**	0.4	0.394
Lighting2	**0.454**	**0.507**	**0.535**	**0.535**	**0.515**	**0.515**
Lighting7	**0.428**	**0.604**	0.564	0.535	**0.354**	**0.35**
OliveOil	**0.637**	**0.698**	**0.604**	**0.719**	**0.604**	**0.774**
OSULeaf	**0.332**	**0.388**	**0.28**	**0.356**	**0.255**	**0.261**
SwedishLeaf	**0.411**	**0.517**	**0.289**	**0.424**	**0.193**	**0.237**
synthetic_control	**0.697**	**0.876**	**0.717**	**0.721**	0.632	0.587
Trace	**0.533**	**0.77**	**0.564**	**0.748**	**0.565**	**0.768**
Two_Patterns	**0.248**	**0.769**	**0.234**	**0.781**	**0.257**	**0.986**
wafer	**0.497**	**0.553**	**0.51**	**0.51**	**0.621**	**0.621**
yoga	**0.337**	**0.364**	0.356	0.379	**0.389**	**0.391**

As the results shows, the quality of MTC is superior for most of datasets using different schemes of clustering.

APPENDIX H

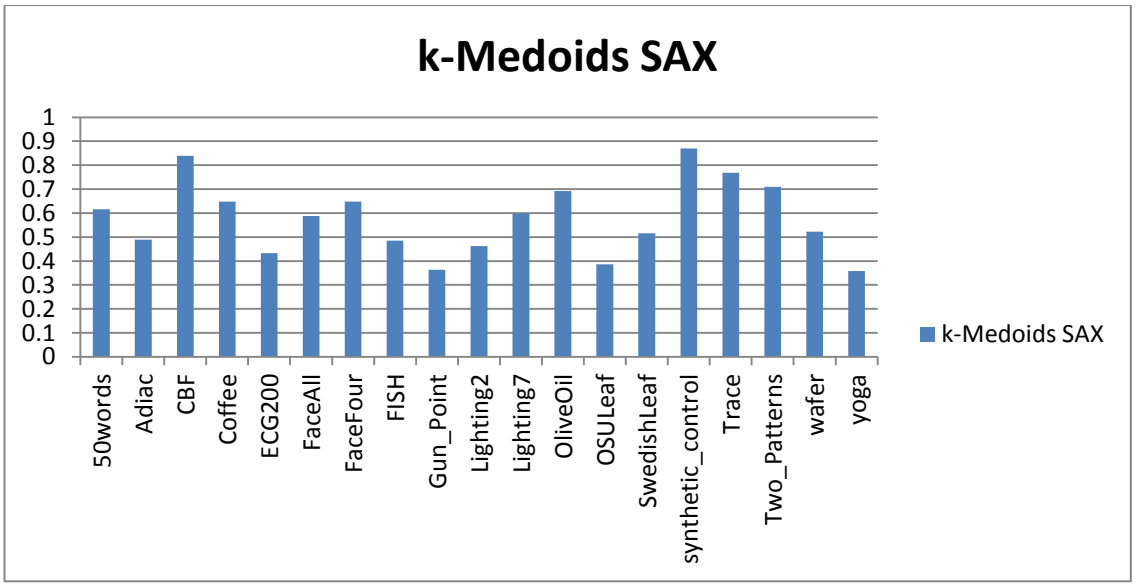
To compare the approach proposed in (X. Zhang et al., 2011) (so called graph-based) with the MTC model, the maximum quality of MTC clustering in front of different orders of nearest neighbour network is calculated. To provide fair conditions, the order of 2 to 5 is considered for graph-based approach which provides reasonable reduction in the second layer. To make sure that this assumption is true for all datasets in hand, the test was repeated for 19 datasets. The reduction ratio of graph-based approach in front of different orders is depicted in the following chart. As this chart shows, order 1 only reduce the data by around 30 % (reduction-rate=0.7), while higher order of nearest-neighbor provides around 70% reduction (reduction-rate=0.3). It is the main reason which we use the order 2 and upper than that for comparison purpose.



Accuracy of merging algorithm using arbitrary shape clustering approach

APPENDIX I

The accuracy of merging sub-clusters using arbitrary shape clustering and multi-prototype approach.



The quality of MTC using arbitrary shape algorithms for merging the prototypes

APPENDIX J

In the following sub-sections, the result of running the MTC model on some datasets is reported in order to show its steps. The datasets used in this experiment are from UCR dataset (TRAIN set).

Leaf dataset

Leaf dataset explained in Section 5.2, is used here for the experiment. All three steps of MTC are applied on this sample dataset to show the results. In the following table the results (reported by implementing MTC in Matlab) related to clustering of the leaf dataset is shown. The report shows complete process and parameters for each step pertaining to a sample run of MTC in order to show the results in each step. It includes three steps. In the first step, the quality is usually low because of approximate clustering. The second step shows the sub-clusters and average affinity (as a pair of average similarity and standard deviation) of each sub-cluster in the first step and second step. As it is indicated, usually, the average similarity increased in the second step. In the third step, using an arbitrary algorithm (e.g., k-Medoids), the prototypes are clustered, and then, the results are evaluated in front of ground truth. The accuracy is shown as average of various indices as quality of final clusters.

The processing of clustering of Leaf dataset using MTC

results
Step 1 Algo:k-Modes | DS:200 | k:6 | dis_method:APXDIST | bound:0 | rep:SAX | alphabet_size:8 | compression_ratio:6 --> Number of clusters:6 | quality:0.34459 | error_rate:12.805 Step 2 DS:200 | dis_method:ED | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 --> Pre-cluster#1 Mems:34 Clus:7 avg_sim_l1:(0.26521-0.044509) avg_sim_DTW:0.44455-0.04864) --> Pre-cluster#2 Mems:29 Clus:7 avg_sim_l1:(0.36488-0.09059) avg_sim_DTW:0.56717-0.089608) --> Pre-cluster#3 Mems:33 Clus:10 avg_sim_l1:(0.30831-0.059757) avg_sim_DTW:0.51149-0.082026) --> Pre-cluster#4 Mems:53 Clus:7 avg_sim_l1:(0.36495-0.091888) avg_sim_DTW:0.59228-0.079599) --> Pre-cluster#5 Mems:36 Clus:11 avg_sim_l1:(0.32262-0.035193) avg_sim_DTW:0.51756-0.08716) --> Pre-cluster#6 Mems:15 Clus:7 avg_sim_l1:(0.26329-0.068317) avg_sim_DTW:0.36841-0.079385) --> Number of clusters:49 | error_rate:0 | reduction:0.755 | correct_rate:1 | purity:1 Making prototype ... Step 3

DS:49 | K:6 | dis_method:DTW | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 clustering:k-medoids --> quality:0.42777

Quality of MTC clustering on Leaf datasets

RI	ARI	Purity	CSM	ConEntropy	F-measure	Jacard	FM	NMI	quality
0.78	0.24	0.56	0.39	0.38	0.53	0.23	0.37	0.37	0.43

Face dataset

The similar process can be seen for Face dataset in the following tables.

The processing of clustering of Face dataset using MTC

Results
Step 1 Algo:k-Modes | DS:24 | k:4 | dis_method:APXDIST | bound:0 | rep:SAX | alphabet_size:8 | compression_ratio:4 --> Number of clusters:4 | quality:0.5233 | error_rate:1.4167 Step 2 DS:24 | dis_method:ED | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 --> Pre-cluster#1 Mems:6 Clus:3 avg_sim_l1:(0.2359-0.098102) avg_sim_DTW:0.25594-0.10573) --> Pre-cluster#2 Mems:4 Clus:1 avg_sim_l1:(0.20996-0.13346) avg_sim_DTW:0.20323-0.14952) --> Pre-cluster#3 Mems:5 Clus:3 avg_sim_l1:(0.25888-0.086967) avg_sim_DTW:0.32599-0.13888) --> Pre-cluster#4 Mems:9 Clus:4 avg_sim_l1:(0.32286-0.066735) avg_sim_DTW:0.32413-0.088161) --> Number of clusters:11 | error_rate:0.15833 | reduction:0.54167 | correct_rate:0.77083 | purity:0.83333 Making prototype ... Step 3 DS:11 | K:4 | dis_method:DTW | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 clustering: hier_avg --> quality:0.60685

Quality of MTC clustering on Face datasets

RI	ARI	Purity	CSM	ConEntropy	F-measure	Jacard	FM	NMI	quality
0.73	0.4	0.71	0.68	0.56	0.78	0.41	0.59	0.59	0.61

Gun point dataset

The following tables show the running of MTC for Gun point dataset.

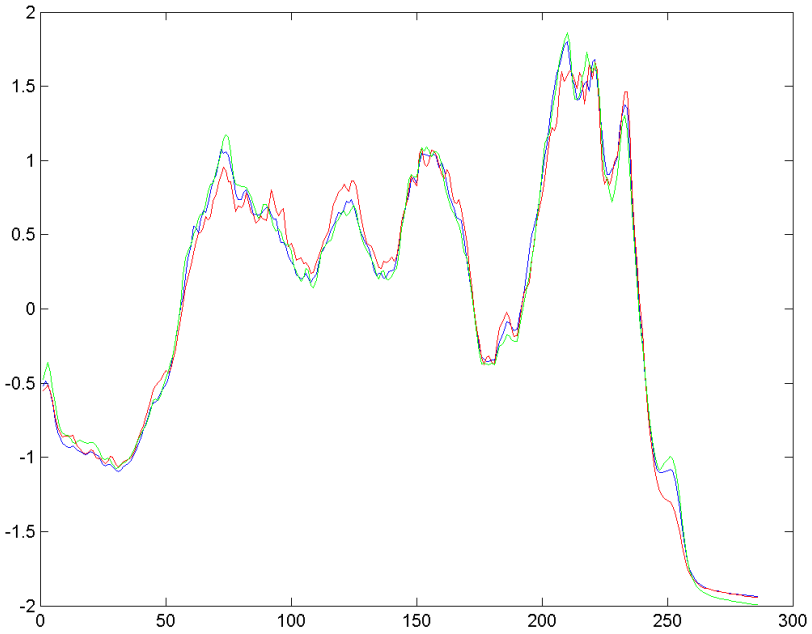
The processing of clustering of Gun point dataset using MTC

Results
Step 1 Algo:k-Modes | DS:50 | k:2 | dis_method:Euclid | bound:0 | rep:SAX | alphabet_size:8 | compression_ratio:6 --> Number of clusters:2 | quality:0.39046 | error_rate:6.08 Step 2 DS:50 | dis_method:ED | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 --> Pre-cluster#1 Mems:25 Clus:7 avg_sim_l1:(0.49478-0.10146) avg_sim_DTW:0.68905-0.12435) --> Pre-cluster#2 Mems:25 Clus:6 avg_sim_l1:(0.58842-0.10705) avg_sim_DTW:0.6277-0.13558) --> Number of clusters:13 | error_rate:0.24 | reduction:0.74 | correct_rate:0.85465 | purity:0.88 Making prototype ... Step 3 DS:13 | K:2 | dis_method:DTW | dtw_bound:1 | rep:RAW | alphabet_size:8 compression_ratio:2 clustering:k-medoids --> quality:0.48283

Quality of MTC clustering on Gun point datasets

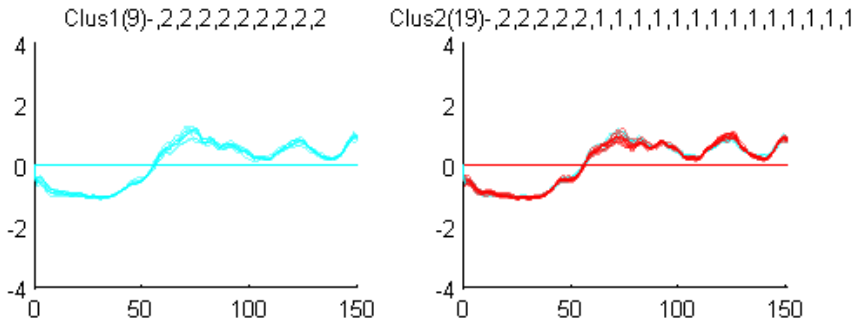
RI	ARI	Purity	CSM	ConEntropy	F-measure	Jacard	FM	NMI
0.54	0.09	0.66	0.62	0.12	0.73	0.41	0.59	0.59

Coffee train dataset



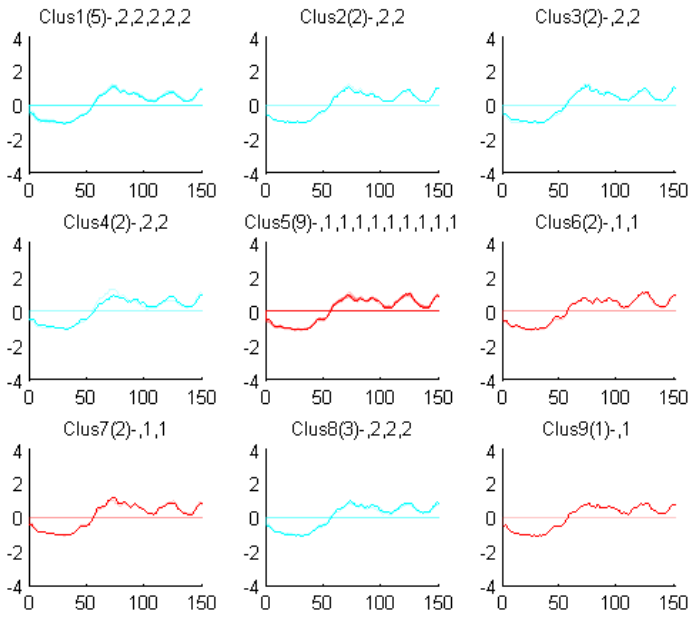
Samples of time-series of coffee dataset

In the first step of algorithm the time-series data are transferred in SAX representation. (SAX(8,6)). Then, k-Modes algorithm is applied on data. The result of first step is depicted in the following chart.



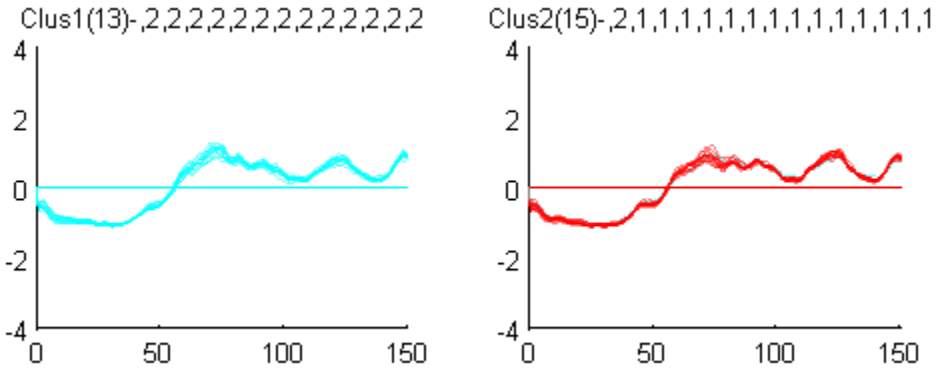
First step of running MTC on Coffee dataset

In the second step, each cluster is decomposed in more pure clusters, using PCS algorithm explained in Section 4.4.2.2.



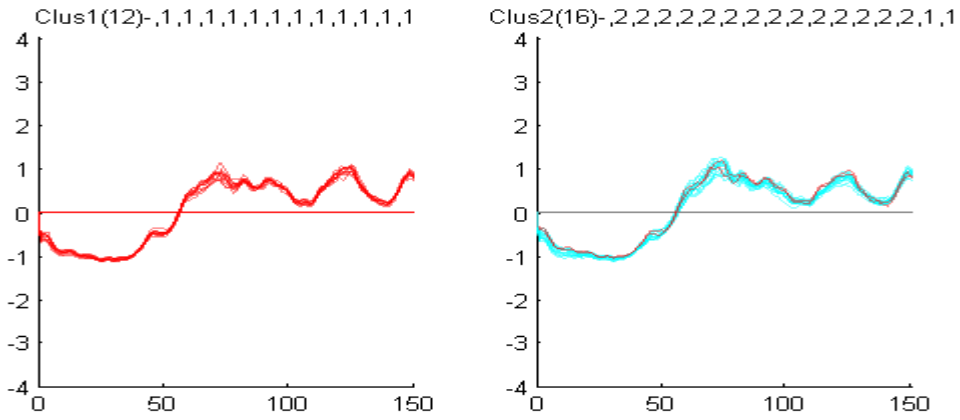
The second step of MTC carried out on Coffee dataset

The result of third step of algorithm is shown in following figure. In this step hierarchical clustering is used to merge the prototypes.



Result of applying MTC (Hierarchical) on Coffee dataset

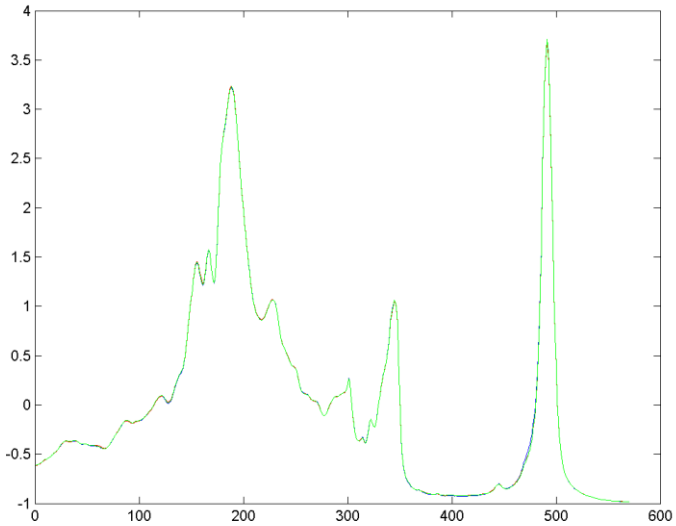
Moreover, the result related to MTC after applying all the steps is repeated using k-Medoids.



Result of applying MTC (Partitioning) on Coffee dataset

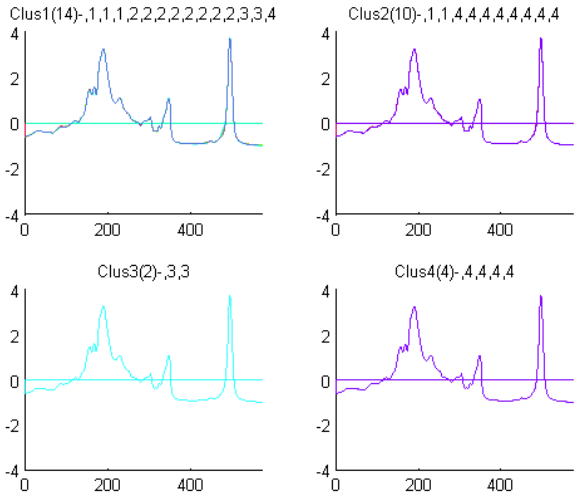
Olive dataset

Samples of datasets of Coffee and Olive oil

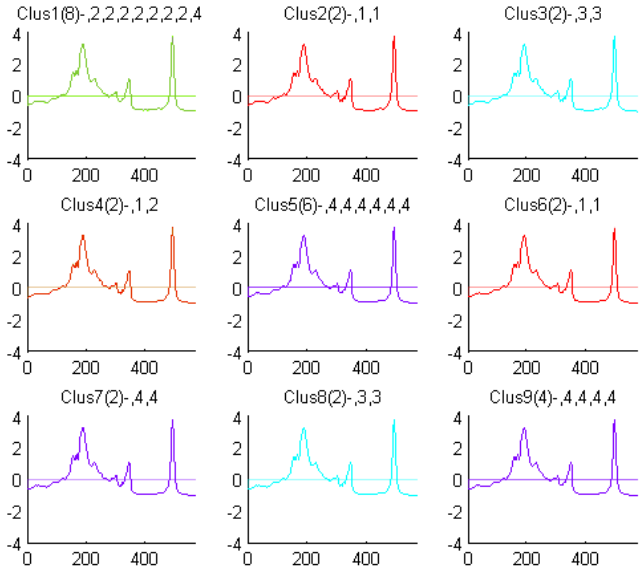


Samples of time-series of olive-oil dataset

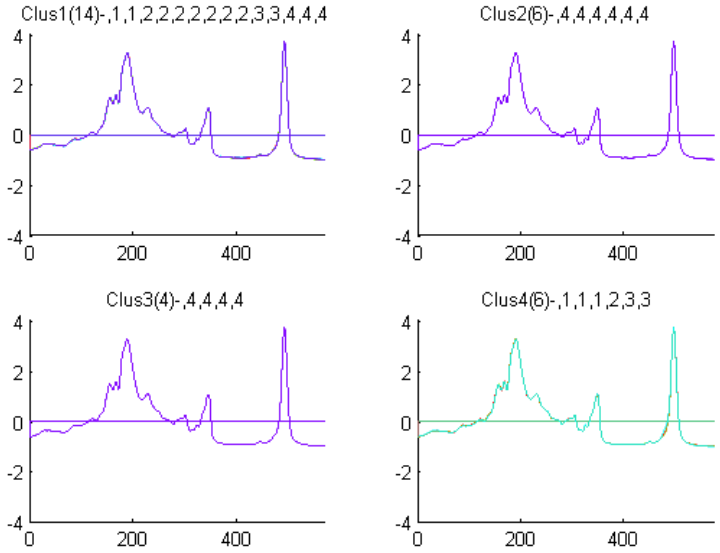
In the same process, MTC is performed on Olive dataset as another sample dataset. Following charts shows the results in each step of MTC.



Results related to first step of running MTC on Olive dataset



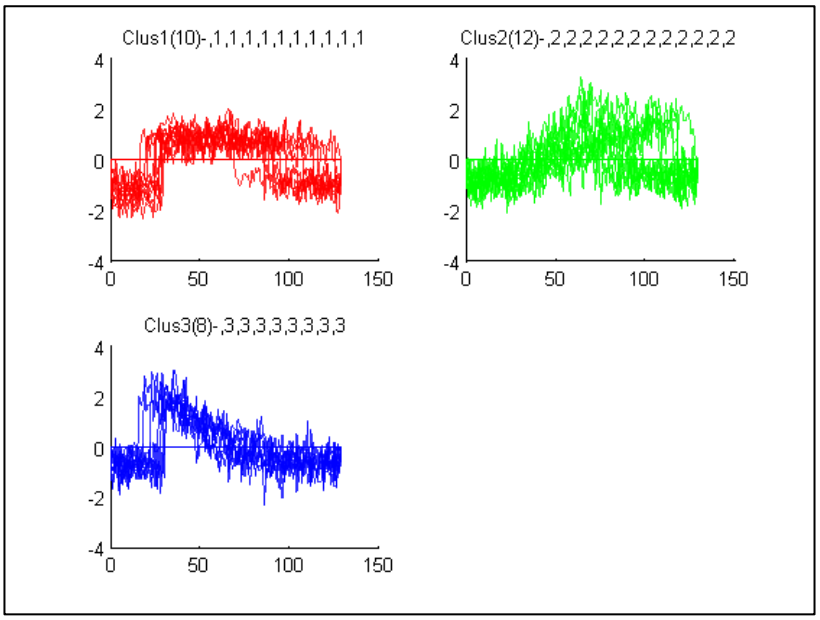
Clusters generated in the second step of MTC on Olive dataset



Results of third step of MTC related to Olive dataset

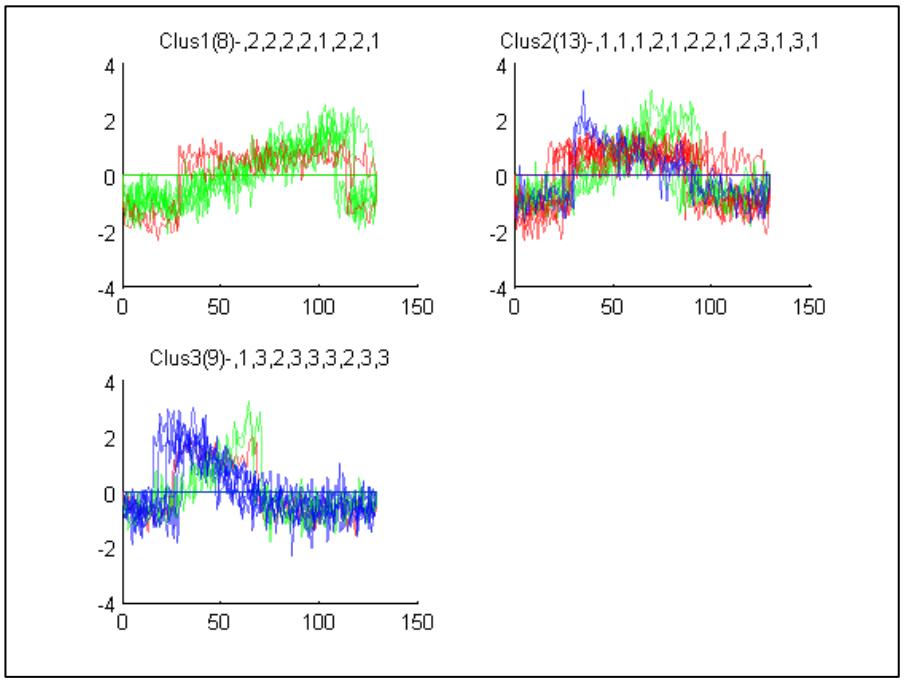
CBF dataset

Cylinder-Bell-Funnel (CBF) time-series in the UCR datasets, is used for the experiment. All three steps of MTC are applied on this sample dataset to show the results visually. Following figures shows the correct answer (ground truth) of CBF datasets.



The ground truth (labels) related to CBF dataset (with 30 instances)

In the first step of MTC, k-Modes clustering is applied on time-series representing by SAX(8,6). The distance measure used for this step is APXDIST. The result of the first step is shown in the following figure. The time-series in the different clusters are represented using a combination of different colors. That is, time-series that belong to the same cluster have all the same color.



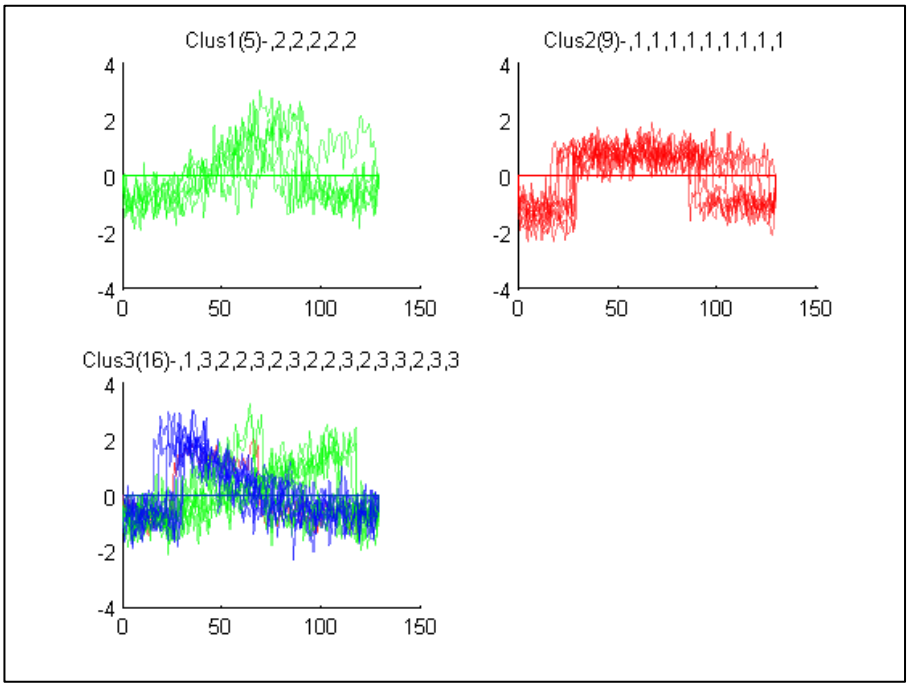
k-Modes clustering of represented time-series using SAX(8,6)

The accuracy of this step is as following table.

Quality of k-Modes clustering on datasets representing by SAX(8,6) on CBF_train dataset

RI	Purity	CSM	Enrtropy	f-measure	jacard	FM
0.63	0.63	0.50	0.24	0.65	0.27	0.43

After applying the MTC model on the CBF dataset, depend on the used merging algorithm in the last step, different results are gained. For example, the following diagram depicts the result of partitioning algorithm using k-Means. Notice that the prototype of each cluster is made similar to centroids of simple objects.



Running MTC on CBF with the k-means algorithm as the last step

The experiment was repeated using various merging approaches in the third step of MTC. The average quality of 100 run is measured to report the quality. All of the runs' results have better quality than first step.

Accuracy of MTC in front of ground truth across various algorithms in the third step

Algorithm	RI	Purity	CSM	Enrtropy	f-measure	jacard	FM
MTC-hier-com	0.66	0.67	0.67	0.40	0.84	0.42	0.60
MTC-k-means	0.74	0.73	0.72	0.57	0.83	0.45	0.63
MTC-hier-avg	0.74	0.73	0.77	0.59	0.90	0.52	0.70
MTC-k-medoid	0.69	0.67	0.70	0.45	0.78	0.45	0.64
MTC-hier-sing	0.73	0.70	0.77	0.56	0.91	0.53	0.71

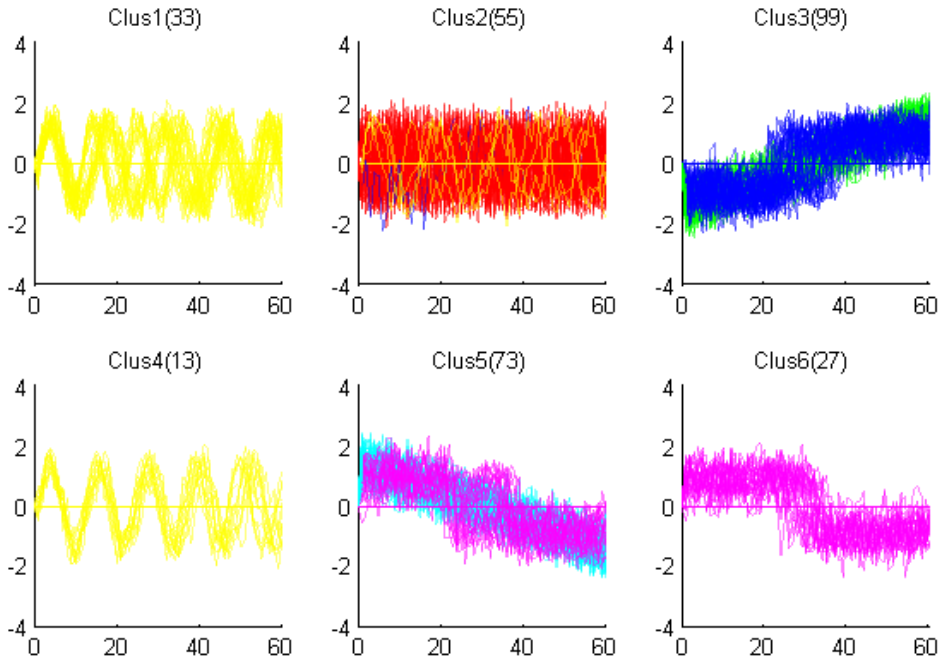
Control Chart dataset

As another experiment, MTC is applied on the Control Chart dataset (CC) as well. In the following table the results (reported by implementing MTC in Matlab) related to clustering of CC is shown. The report shows complete process and parameters for each step pertaining to a sample run of MTC.

The results related to clustering of CC using MTC

```
Results:
step 1
 dis_method:SAXAPX rep:SAX alphabet_size:8 compression_ratio:6
 --> DS:300 clusters:6
 --> quality:0.74968
step 2
 dis_method:DTW dtw_bound:0.5 rep:SAX alphabet_size:8 compression_ratio:6
 --> cluster:1 Mems:33 Clus:3
 --> cluster:2 Mems:55 Clus:4
 --> cluster:3 Mems:99 Clus:7
 --> cluster:4 Mems:13 Clus:4
 --> cluster:5 Mems:73 Clus:5
 --> cluster:6 Mems:27 Clus:5
 Number of clusters:28
step 3
 dis_method:DTW dtw_bound:0.5 rep:SAX alphabet_size:8 compression_ratio:2
 Making prototype ...
 clustering:k-Medoids
 --> quality:0.89394
```

The first step of clustering is shown in the following chart for more intuition.



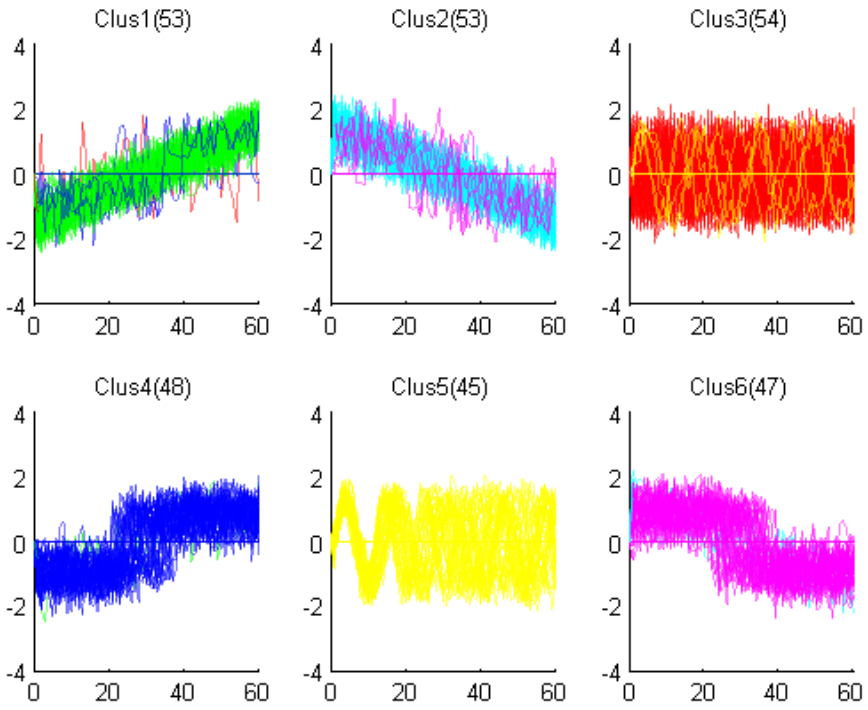
The first step of clustering of CC using MTC

Moreover, the accuracy gained after running the first step is reported in following table.

Accuracy of k-Modes clustering on datasets representing by SAX(8,6) on CC dataset

RI	Purity	CSM	Enrtropy	f-measure	jacard	FM
0.88	0.74	0.75	0.75	0.85	0.54	0.71

After running the algorithm, the final result is shown in following. The result illusterate how the cluster purity is increased. Accordingly, the accuracy calculated at the end of the clustering is reported in the following table.



Running MTC on CC with the k-Medoids algorithm as the last step

RI	Purity	CSM	Enrtropy	f-measure	jacard	FM
0.96	0.94	0.88	0.87	0.94	0.79	0.88