5-2013

Rank Based Anomaly Detection Algorithms

Huaming Huang
Syracuse University

ABSTRACT

Anomaly or outlier detection problems are of considerable importance, arising frequently in diverse real-world applications such as finance and cyber-security. Several algorithms have been formulated for such problems, usually based on formulating a problem-dependent heuristic or distance metric. This dissertation proposes anomaly detection algorithms that exploit the notion of “rank," expressing relative outlierness of different points in the relevant space, and exploiting asymmetry in nearest neighbor relations between points: a data point is “more anomalous" if it is not the nearest neighbor of its nearest neighbors. Although rank is computed using distance, it is a more robust and higher level abstraction that is particularly helpful in problems characterized by significant variations of data point density, when distance alone is inadequate.

We begin by proposing a rank-based outlier detection algorithm, and then discuss how this may be extended by also considering clustering-based approaches. We show that the use of rank significantly improves anomaly detection performance in a broad range of problems.

We then consider the problem of identifying the most anomalous among a set of time series, e.g., the stock price of a company that exhibits significantly different behavior than its peer group of other companies. In such problems, different characteristics of time series are captured by different metrics, and we show that the best performance is obtained by combining several such metrics, along with the use of rank-based algorithms for anomaly detection.

In practical scenarios, it is of interest to identify when a time series begins to diverge from the behavior of its peer group. We address this problem as well, using an online version of the anomaly detection algorithm developed earlier.

Finally, we address the task of detecting the occurrence of anomalous sub-sequences

within a single time series. This is accomplished by refining the multiple-distance combination approach, which succeeds when other algorithms (based on a single distance measure) fail.

The algorithms developed in this dissertation can be applied in a large variety of application areas, and can assist in solving many practical problems.

RANK BASED ANOMALY DETECTION ALGORITHMS

By

Huaming Huang
B.S. Xiamen University, 2000
M.S. Syracuse University, 2009

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer and Information Science and Engineering (CISE)

Syracuse University
May 2013

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 What are outliers?

Outlier detection techniques attempt to find the objects that are “different" from the rest of the data objects in a given data set. Usually, outliers are generated from certain system mechanisms that are very different from the system mechanism of the rest of data set. The problem of outlier detection is of considerable importance, arising frequently in many different domains such as fraud detection, cyber-intrusion detection, medical anomaly detection, image processing and textual anomaly detection [11]. In financial area, banks spend millions of dollars on detecting credit card fraud and money laundering; using outlier detection techniques, they wish to identify abnormal usage patterns as soon as possible in order to prevent the future loss. In cyber-intrusion field, companies use outlier detection techniques to identify a hacker’s attacks by analyzing the computer or website log files, and then attempt to identify abnormal user behaviors. In image processing, users can take advantage of outlier detection algorithms to identify the image consisting of different objects.

Many researchers have attempted to describe an anomalous object in a given data set. For example, Hawkins [29] suggests that "An outlier is an observation that deviates so

much from other observations as to arouse suspicion that is was generated by a different mechanism". Similarly, Chandola *et al.* [11] say that "Anomalies are patterns in data that do not conform to a well defined notion of normal behavior". But in real-world applications, a well defined normal behavior sometimes is hard to identify and it may also dynamically change. For instance, a normal user's behavior on a computer may not be the same as another normal user's behavior, and all behaviors can change along the time: in the spring semester, students spend more time on preparing homework, document editing, research, whereas in the summer, they spend more time on movies, online games and so on.

Knorr and Ng [43] use very specific approach to define an outlier as "An object p in a data set D is a DB(pct, dmin)-outlier if at least pct percent of the objects in D lie greater than distance $dmin$ from p." This definition is a typical definition for distance-based outlier detection algorithms, but it doesn't capture all kinds of outliers. For instance, in Figure 1.1, if the object 'a' is considered as an DB(pct, dmin)-outlier then according to this definition all objects in cluster C2 are also considered as DB(pct, dmin)-outliers, which is counter intuitive; since object 'a' looks more suspicious than all objects in cluster C2.

To overcome the deficiency of the above definition, Breunig *et al.* [5] suggest to use local outlier factor (LOF) to capture the degree of an outlier, which is essentially the average of the ratio of the local reachability density of an object and those of the object's k nearest neighbors (kNN). The value of LOF indicates the degree of the oulierness of an object; LOF > 1 generally indicates that the object is an outlier, whereas if LOF is1 or less than the object is a non-outlier. LOF is also one of density-based outlier detection algorithms. Tang *et al.* [59] obtain the connectivity-based outlier factor (COF) to capture the outlierness of an object. Jin *et al.* [36] assign to each object the degree of being influenced outlierness (INFLO) and introduce a new idea called 'reverse neighbors' of a data point when estimating its density distribution. The common theme among these algorithms is that they all assign outlierness to each object in the data set and an object will be considered as an outlier if its outlierness is greater than a pre-defined threshold (usually the threshold is determined by

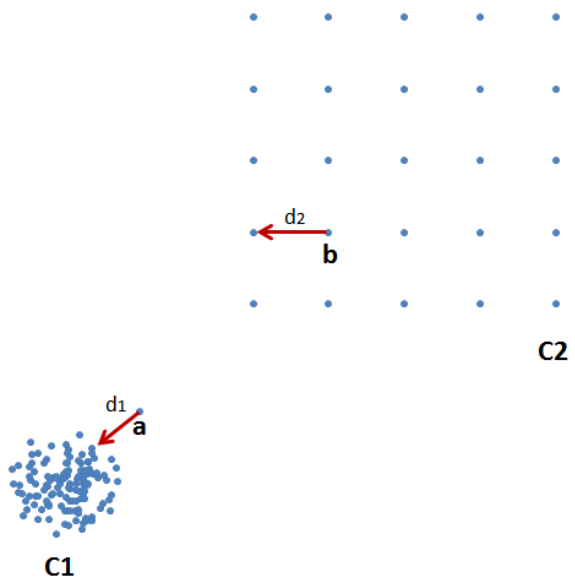

Fig. 1.1: **A case that DB(pct, dmin)-outlier definition and distance-based method doesn't work.** - There are two different clusters C1 and C2, and one isolated data object 'a' in this data set. The distance from object 'b' to its nearest neighbor, d_2 , is larger than d_1, the distance from object 'a' to its nearest neighbor, which makes 'a' unable to be identified.

users or domain experts). The density-based definition seems to work better, but it also has deficiencies. For example, if an outlier has neighbors from different density clusters, then its outlierness is measured incorrectly, as seen in Figure 1.2.

So, in order to capture the real anomalousness of a true outlier, more precise and reasonable definitions and algorithms for outlier detection are developed in this dissertation.

This chapter is organized as follows: Section 1.2 introduces the types of outliers and Section 1.3 shows our general goals. Sections 1.4, 1.5 and 1.6 present existing point outliers detection techniques, time series outlier detection techniques and existing online detection techniques respectively.

1.2 Types of Outliers Detection Problems

There are many outlier problems in real world applications; Chandola *et al.* [11] group them into three categories:

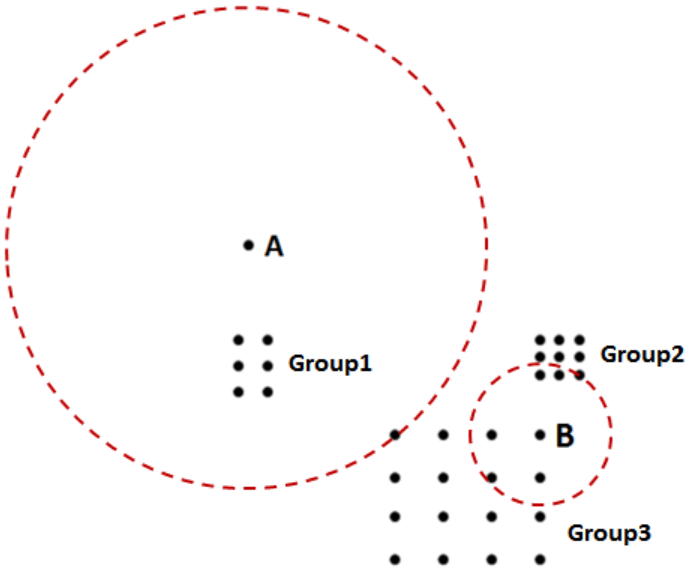

Fig. 1.2: **Illustration of a case that density-based algorithm would not work.** - Red dash circles contain the k nearest neighborhood of 'A' and 'B' when k=7.

- **Point outliers** – Individual data objects that are distinct with respect to the rest of data set.

- **Contextual outliers** – If data objects are considered as anomalous in a specific context but not in other situations, then they are called contextual outliers.

- **Collective outliers** – If a set of data objects is considered as anomalous with respect to the entire data set, then members of the set are called collective outliers.

These three types cover many types of outliers, however, they cannot precisely cover all. The types of data set must be first introduced. Based on data set's characteristics, there are two types of datasets: set of static data points and time series dataset. The key difference between these two types is time stamp. Both are multi-dimensional; both have dependencies but in time series the same feature is time stamp.

- Static data set simply means a collection of data objects (real-valued) without any time feature or any order by time. In other words, in static dataset the data set doesn't

change with time. Static dataset could be multivariate (multiple attributes) or univariate (single attribute).

- Time series data set represents a collection of data objects with time feature or ordered by timestamps. In this dissertation, we only consider real-valued time series.

In this thesis, we address three specific types of outlier detection problems.

- **Point Outliers Detection**. For static data set, point outlier detection is our focus. Note that a data set may contain more than one outlier. For example, in Figure 1.3, data objects "A, B, C and D" are considered as point outliers since they are not in any cluster and are far away from the majority of the data objects. So they are different from the rest of data objects using the 'neighborhood' aspect. Point outliers detection has attracted much attention in practical applications such as detecting criminal activities, fraud detection, and exceptional cases. In these areas, rare cases may be more interesting and meaningful than normal cases.

- **Contextual Abnormal Subsequence Detection**. If a collection of data objects of a series is considered as anomalous with respect to the entire single series, then it is called contextual abnormal subsequence. This anomaly is for a time series data set. For instance, in Figure 6.4, the subsequence highlighted by red is obviously anomalous compared with all other subsequences in the series. For contextual abnormal subsequences, we are more interested in detecting the positions of anomalies as soon as possible. Contextual abnormal subsequence detection is a very interesting problem for detecting hacker attacks for computers, or detecting a premature ventricular contraction (PVC) in ECG (electrocardiograms) series.

- **Anomalous Series Detection (Collective Outliers)**. If a single time series is anomalous with respect to the other series in a given data set, then it is called an anomalous

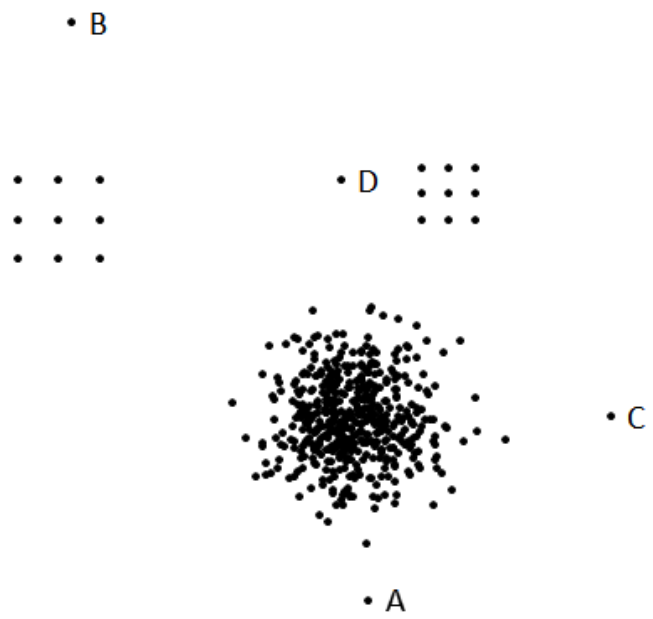

Fig. 1.3: **Outliers and a static data set** - Data point "A,B,C and D" are outliers with respect to the rest of data objects in this two dimensional data set.

Table 1.1: Outlier Problems and Corresponding Data sets

Outlier Types	Data set Types
Point Outliers	Static
Contextual Abnormal Subsequences	Time Series
Anomalous Series	Time Series

series. Anomalous series may also contain point outliers and contextual abnormal subsequences. Two anomalous series are shown in Figure 1.5. Anomalous series detection can be applied for detecting unusual patterns or special events within time series data set such as detecting financial fraud in credit card dataset [23], and detecting light curves' outliers within astronomical data [53].

Table 1.1 shows the types of outliers and their corresponding data sets.

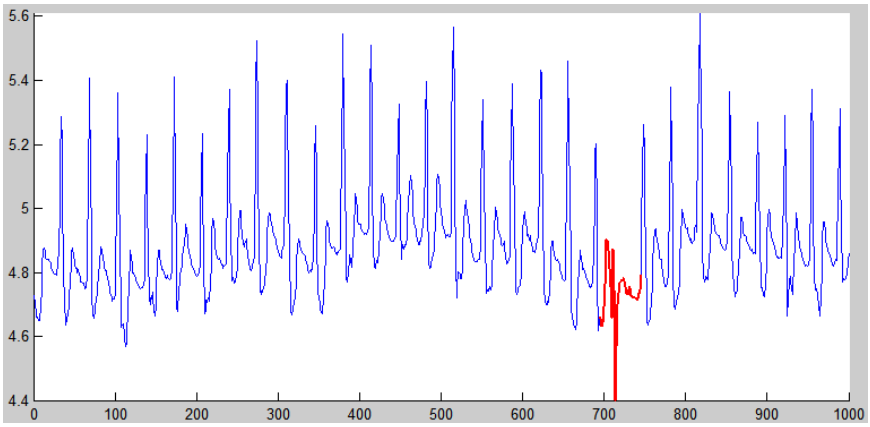

Fig. 1.4: **Illustration of contextual abnormal subsequence.** - A subset of ECG data. Red line represents abnormal signal sequence.

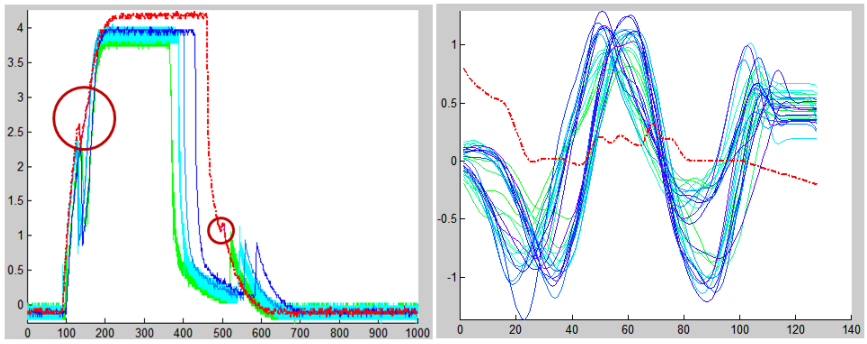

Fig. 1.5: **Examples of anomalous time series** - Red dash represents anomalous series. (Left) Anomalous time series with abnormal subsequences, red circles show the abnormal subsequences. (Right) Another example of anomalous time series.

1.3 Our Goals

Our main research will focus on the three outlier detection problems just discussed and our goal is to develop algorithms which have the following characteristics:

1. Normal behaviors have to be dynamically defined. No prior training data set or reference data set for normal behavior is needed.

2. Outliers can be effectively detected even if the distribution of data is unknown.

3. Be adaptive. It can be applied or modified for outlier detection in different domains.

4. Less domain knowledge required of users.

By following above targets, we propose a series of rank based algorithms including rank based detection algorithms, clustering and rank based detection algorithms, multiple measures based anomalous series detection algorithms, and its extended version for online detection. We also propose an algorithm for abnormal subsequence detection.

.

1.4 Point Outlier Detection for Static data set

Point outliers in the static data set usually have abnormal attributes with respect to the rest of data set. Formally, given a data set $\mathcal{D}$, $p \in \mathcal{D}$, our goal is to find $O(p)$, outlierness of p, for each object and $O_{threshold}$, such that if $O(p) \geq O_{threshold}$, then p is an outlier, otherwise not.

For static dataset, the most important task is to identify one or more individual anomalies within a given data set. For example, in Figure 1.3, four isolated data points "A,B,C and D" are far away from the rest of data points, hence, are point outliers.

Unlike supervised data mining techniques, outlier detection is typically an unsupervised learning problem because the behaviors or patterns of outliers are unknown. In some

real world application domains, algorithms may need domain expert's help to decide the optimized parameters, but in most cases this is not possible. Thus, practical applications require outlier detection techniques to make as few assumptions as possible and have to be adaptive to the varieties of anomalies.

1.4.1 Existing Approaches

In recent decades, researchers have proposed different approaches to detect anomalies. Statistics-based approaches (see [30, 51]) were first used for outlier detection based on the assumption that the distributions of datasets are known, e.g., Gaussian distribution or Uniform distribution. A data point was defined as an outlier if it deviates from the existing distribution over a threshold such as three standard deviations. With sufficient knowledge about the dataset, statistics-based methods work effectively. But in the real world, distributions of the data are unknown or arbitrary, significantly impacting the performance of such methods. To overcome this obstacle, clustering-based algorithms have been proposed to detect outliers [27, 69]. The basic idea is that a data point is an outlier if it does not belong to any cluster. First, these methods build up the clusters by using different algorithms and criteria, then outliers can be found by removing all points that belong to clusters. The effectiveness of this approach depends on the clustering algorithm. Unfortunately, most clustering algorithms are designed for detecting clusters instead of outliers, so the selection performances varies for different data sets.

Knorr and Ng [43] propose to detect an outlier based on its distances from neighboring data points; many other variations of distance-based approaches have been discussed in the literature[19, 20, 70]. The main deficiency found in distance-based approaches is the assumption that the distance from an outlier to its neighbors exceeds the distance from a non-outlier to its neighbors, which is not always true. For example, in Figure 1.1, there are two clusters C1 and C2 of different size. The distance from object 'b' to its neighboring objects is apparently greater than the distance from 'a' to its neighboring objects which

leads to a wrong conclusion that ‘b’ is more suspicious of an outlier than ‘a’. Obviously, ‘a’ is more suspicious of an outlier than ‘b’ since ‘a’ is far away from all its neighbors and ‘b’ is closer to its neighbors. The closeness cannot be simply measured by distance alone.

In order to overcome the deficiencies of distance-based methods, Breunig *et al.* [5] proposed that each data point of the given data set should be assigned a degree of outlierness. In their view, as in other recent studies, a data point’s degree of outlierness should be measured relative to its neighbors; hence they refer to it as the “local outlier factor" (LOF) of the data point. Tang *et al.* [59] argued that an outlier doesn’t always have to be of lower density and lower density is not a necessary condition to be an outlier. They modified LOF to obtain the “connectivity-based outlier factor" (COF) which they argued is more effective when a cluster and a neighboring outlier have similar neighborhood densities. Local density is generally measured in terms of k nearest neighbors; LOF and COF both exploit properties associated with k nearest neighbors of a given object in the data set. However, it is possible that an outlier lies in a location between objects from a sparse and a denser cluster. To account for such possibilities, Jin *et al.* [36] proposed another modification, called INFLO, which is based on a symmetric neighborhood relationship. That is, their proposed modification considers neighbors and ‘reverse neighbors’ of a data point when estimating its density distribution. Tao and Pi [60] have proposed a density-based clustering and outlier detection (DBCOD) algorithm, which also belongs to the density-based algorithms.

These density-based outlier detection algorithms, such as LOF, COF and INFLO, use the following methodology:

- Define the concept of density of a data point; using the notion of neighborhood (or some variation of it); and
- Calculate the “outlierness” of an object; usually defined as the ratio of a data point’s density with the density in the region surrounding the data point.

1.4.2 Drawbacks in Density-based Approaches

Density-based methodology that exploits k-neighborhood of a data point has many good features. For instance, it is independent of the distribution of the data and is capable of detecting isolated objects. However it also has some shortcomings:

- Density-based algorithms assume that all neighborhoods of a data point have similar density. If some neighbors of the point are located in one cluster, and the other neighbors are located in another cluster and the two clusters have different densities, then comparing the density of the data point with all of its neighbors may lead to a wrong conclusion and the recognition of real outliers may fail, an example is shown in Figure 1.2.

- The notion of density does not work well for sparse data sets such as a cluster of points on a single straight line. Even if each point in the set has equal distances between itself with its closest neighbors, it may still have different density depending on its position in the dataset.

1.4.3 Our Solution

Before proposing a solution for anomaly detection problem, let us consider a concrete example. Consider the case where a financial institution wishes to find anomalous behavior of its customers. It is obvious that all normal customers are not alike – depending upon their needs and nature all have different 'normal' behavior. Individuals with multiple accounts and large investments tend to have deposit and withdrawal patterns much different compared to the individuals with small amounts. Consequently, collection of data associated with *all* users is likely to form multiple clusters with variable number of data points in each cluster, variable densities, etc. In this data set no two clusters look alike, although they consist of 'normal' behavior.

In such situations, to find anomalous observations, the ideal solution is to transform the

data so that in the transformed space all data points have the same statistical distributions. In this dissertation we have attempted to achieve this goal via ‘ranks’ and ‘modified ranks’. In using rank based approach (especially local to the object of interest) we manage to diminish the effect of inter-object distances; and in using ‘modified-ranks’ we diminish the effect of the size of the local cluster(s).

We propose several new approaches [31, 32] for outlier detection, based on a ranking measure and clustering that focuses on the question of whether a point is “interior" for its nearest neighbors. Using our methods, low cumulative rank implies the point is central. For instance, a point centrally located in a cluster has a relatively low cumulative sum of ranks because it is among the nearest neighbors of its own closest neighbors; however a point at the periphery of a cluster has a high cumulative sum of ranks because its nearest neighbors are closer to each other than the point. Use of ranks eliminates the problem of density calculation in the neighborhood of the point, and this improves performance. Our method performs better than several density-based methods, on some synthetic data set as well as on some real data set. More details will be discussed in the following chapters.

1.4.4 Our Contributions

We have implemented novel algorithms based on ranks. Although based on distance, ranks work better than distance and our method captures the anomalousness of an object more precisely in most cases; and it can be applied in broad practical domains.

Rank based clustering algorithm can be used for detecting tight and sparse clusters easily using only one parameter. Based on ranks, it forms clusters with higher accuracy. By combining it with ranks, we show that better performance can be achieved than most of algorithms we have compared with.

1.5 Outlier Detection for Time Series Data Sets

Anomalous series detection and contextual abnormal subsequence detection are both applicable for time series data set. In the current research, only real-valued time series are considered, categorical-valued time series are out of the scope of this investigation. Anomalous series detection only focuses on identifying anomalous series whereas the contextual abnormal subsequence problem requires that we detect abnormal subsequence within a single series, and it requires the comparison between subsequences and the rest of the series. The main difference between these two techniques is: the first one tries to find out which series is anomalous while the latter one tries to find out when abnormal behaviors occur.

These two problems are far more complicated than the algorithms for point outlier detection because of the following challenges:

- Historical information of a series must be considered, but how to summarize the useful historical information is a tough problem.
- Defining a “normal series" is difficult. Since the data in each time series is changing with time, so a normal series at $t = 1$ may become anomalous at $t = 2$ which makes finding normal series more difficult.
- The behavior of outliers is different for different applications, and it makes detecting abnormal behavior a tough task.
- The noise in the normal series may also cause misleading answers.
- Even in a single application domain, the outlier is also changing with time, so it requires any effective algorithms or techniques to be very adaptive and flexible to deal with dynamic detection. The algorithm must be sensitive to dynamically changing outliers.

.

On-line detection is another important problem. When time series data is huge, getting useful results in a short time using large amount of data is really a big challenge.

These are challenging problems that researchers have to deal with. In the last decade, researchers have proposed different procedures to deal with these issues. Precise problem definitions and associated existing techniques will be discussed in the following sections, along with a brief introduction to our approaches.

1.5.1 Anomalous Time Series Detection

Anomalous time series detection, or outlier time series detection, is an important task in data mining for time series data set, especially for detecting unusual patterns or special events within time series data set Examples include detecting financial fraud in credit card dataset [23], detecting light curves' outliers within astronomical data [53], and detecting shape anomalies [47] [62].

The problem can be formulated as follows. Given a time series data set $\mathcal{D} = \{x_i(t)|1 \leq t \leq n; i = 1, 2, \ldots, m\}$, where $x_i(t)$ represents the data object of the ith time series at time t, n is the length of series and m is the number of time series in the dataset, the goal is to find the outlierness of a series x_i, $O(x_i)$, and the threshold $O_{threshold}$ such that if x_i is an anomalous series, then $O(x_i) \geq O_{threshold}$; otherwise not. Given $\mathcal{D}$, the algorithms should be able to calculate the $O(x_i)$ and $O_{threshold}$ for $\mathcal{D}$ automatically. Here, the meaning of "difference" between anomalous series and normal series is determined by $O_{threshold}$, and it may change from one domain to another, in other words $O_{threshold}$ should not be a fixed value, instead, it needs to be dynamically adjusted for different domains. For example, a threshold to detect a hacker's attacks is obviously different from a threshold to detect a shape anomaly. Sometimes, the same external cause may result in abrupt changes in most of the time series being considered, e.g., increases in stock prices of most defense contractors as a result of an announcement of impending military conflict. This should not be considered as a case of one time series being anomalous with respect to other time

series. The algorithm should able to distinguish this from real anomalous series.

1.5.1.1 Existing Approaches

In recent years, researchers have used many ideas to find an anomalous time series. Leng, et al. [44], use a new time series representation method, based on key points and dynamic time warping to find the anomalous series. Fujimaki *et al.* [25] construct a global AR model for all series and then measure the anomaly score at time t as the gap between the observed value and the predicted value. Keogh and Ratanamahatana [38] propose a lower bound based pruning method for dynamic time warping to detect anomalous series. Protopapas *et al.* [53] uses cross-correlation as the measure of similarity between two individual light curves and use the average correlation as outlier measure. Yankov *et al.* [68] propose fast k-nearest neighbor and early abandon to find the unusual time series for a time series database. Discrete Wavelet Transform [8], and Discrete Fourier Transform [21] also can be applied for anomalous series detection by calculating the distance between coefficients of transformed series.

1.5.1.2 Issues and Our Solution

To the best of our knowledge, most of existing methods for anomaly detection relies on a single measure; however, in general, any individual measure fails to capture varieties of anomalies which may occur in multiple application domains. We propose an anomaly detection method based on a combination of multiple distance measures (MUDIM) [34] where strengths lies in the properties that several types of anomalies can be captured by it.

1.5.1.3 Our Contributions

We show that our multiple distance measures based methods do better than single distance measure based method and it can capture a variety of anomalous series without requiring more domain knowledge. Rank-based algorithms are applied to adjust the weights of dif-

ferent features. The most interesting advantage of it is that in our approach the weights change automatically, based on anomalousness of each distance feature; other algorithms do not achieve the same goal in the same conditions.

1.5.2 Abnormal Subsequences Detection in a Single Series

This problem can be formulated as follows. Suppose $X = \{x(t)|1 \leq t \leq n\}$ is a time series where $x(t)$ represents the value of the series X at time t and n is the length of X. A subsequence of X is defined as $X_{p,m} = \{x(p), ..., x(p+m-1)|1 \leq p \leq n-m+1; 1 < m \leq n\}$. The goal is to find an abnormal subsequence $X_{p,m}$ such that outlierness of $X_{p,m}$ exceeds a specified threshold. An abnormal subsequence is also known as “discord" in other literatures [41, 6, 13].

1.5.2.1 Existing Approaches

To detect any abnormal subsequence in a series many solutions have been proposed. Lin et al. [45] uses SAX, suffix tree and non-self match to find unusual medical time series discords such as ECG (electrocardiograms) data set. ECGs are time series that measure the electrical activities of the heart, and a premature ventricular contraction (PVC) is a relatively common cardiac arrhythmia that can be easily detected in ECG compared with a normal heart beat by a cardiologist. The techniques for anomalous time series detection can be applied here to help cardiologists to identify the status of patient's heart.

Keogh *et al.* use ‘ heuristic discord discovery’ based on augmented tries for detecting discord [37, 41]. Fu *et al.* [13] and Bu *et al.* [6] both suggest a method based on haar wavelet transformation to find the best alphabet size and word size to further improve speed of their time series discord detection algorithm by pruning unnecessary calculations.

1.5.2.2 Our Solutions

We proposed a novel outlier detection method based on sliding window and multiple distance measures. The method achieves better performance than the existing methods in a variety of datasets used in our experiments.

Our algorithm can detect all abnormal subsequences within a single series with only one parameter, w, the length of subsequence. Unlike other kNN based methods, it is able to detect all abnormal subsequences even when k similar subsequences exist in a single series, whereas other methods may fail.

1.6 Online (Real-time) Anomalous Series Detection

Online detection requires detecting the anomalous series as soon as possible. So the algorithms for online detection must be very effective and efficient.

1.6.1 Related Research

This problem has been studied by various researchers [67, 63, 40, 61, 56, 9, 41].

Yamanishi, *et al.* [67], address this problem using a probability-based model in non-stationary time series data. Wei, *et al.* [63], use frequencies of symbolic aggregation representation (SAX) of time series. Keogh, *et al.*, [40, 41] employ suffix tree comparison, HOT SAX algorithm, and nearest non-self match distance. Toshniwal, *et al.*, [61] extend HOT SAX algorithm to detect outliers in streaming data time series. Sadik, et al. [56], develop an algorithm based on adaptive probability density functions and distance-based techniques to detect outliers in the data streams. Xie and Tang [65] propose a new dynamic hidden semi-Markov model to detect time-variant user behavior.

Unfortunately, some of the above online methods can only be applied for detecting outliers in a single series. In contrast with the above methods, our focus is on the important variation where multiple time series are being observed simultaneously, and detection

should occur as soon as one (or more) series begins to differ from the rest. In our approach no training set is required and no model is constructed *a priori*. Some methods mentioned earlier, such as [63, 56], can be used directly for online detection. Other methods, proposed by Chandola *et al.* [10], such as anomalous series detection with training data sets based using kNN, window based, predictive model based, segment based methods, can also be adapted for online detection. But the main problems found in these methods are:

- Finding the optimal values of parameters of the algorithms requires the user to have expert domain knowledge which may not be available.
- High amount of computational effort is required, so that these methods are not suitable for online detection in large data streams.
- Concept drifting (the statistical properties of data objects change over time) and data uncertainty are common problems in data streams, which prevent successful application of fixed model based methods.

1.6.2 Our Solutions

To overcome these problems, we propose online algorithms [33] that utilize information from multiple distance metrics. These unsupervised methods require only one parameter, viz., the number of nearest neighbors (k). Experimental results show that our approaches are successful in anomalous time series detection, and also computationally efficient enough to permit online execution. Our online algorithms have higher accuracy and works better in more data sets than other competing methods.

CHAPTER 2
POINT OUTLIERS DETECTION BASED ON RANKS

The goal for point outliers detection algorithms is to find the point outliers that are “different" from the rest of the data points in a given data set. Practical applications concerning outlier detection occur in many domains such as fraud detection, cyber-intrusion detection, medical anomaly detection, image processing and textual anomaly detection [11].

As mentioned in the first chapter, density-based algorithms seem to work better in most of the application domains, so we will first review typical density-based algorithms in detail. This chapter is organized as follows: in Section 2.1 we review the density based approaches of anomaly detection algorithms. Then in Section 2.2 we introduce our algorithm - RBDA, and finally show the experiment results in Section 2.3.

2.1 Review of Density-based Approaches

In density based approaches the main idea is to consider the behaviors of a point with respect to its neighbors. The neighborhood is conceptualized by considering k nearest neighbors, where k is either iteratively estimated or is a preassigned integer. The underlying

assumption is that if density of a point p is 'different' than densities of its neighbors, it must be an anomaly. The main difference between the approaches described below is in how they define the local behavior related with density.

2.1.1 LOF (Local Outlier Factor) approach

Breunig *et al.* [5] proposed the following approach to find an outlier. For each point p in the given dataset, they evaluate its "local outlier factor" (LOF); a point whose LOF is 'large' is declared anomalous.

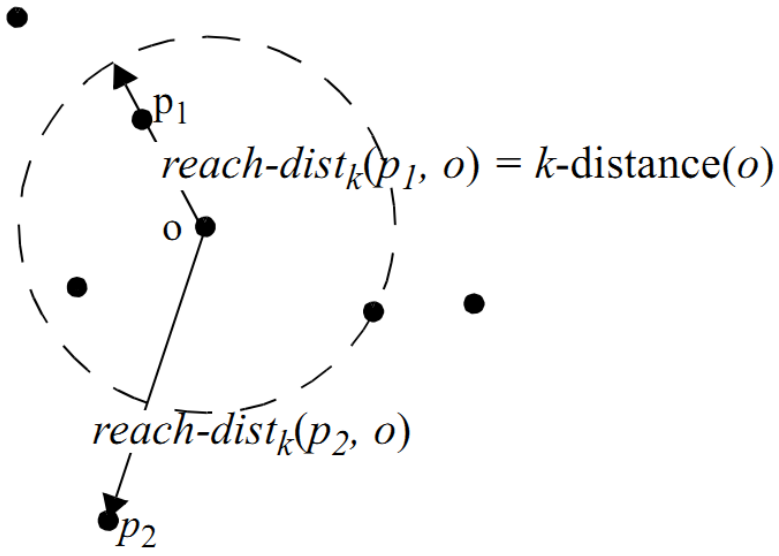

Fig. 2.1: **Illustration of reachability distance.** - $reach - dist_k(p1, o)$ and $reach - dist_k(p2, o)$ for k=4. This figure is from Breunig [5]

1. Find the distance, $d_k(p)$, between p and its kth nearest neighbor. The distance can be any measure; but here the Euclidean distance is used. Denote the set of k nearest neighbors of p by $\mathcal{N}_k(p) = \{q \in D - \{p\} : \ d(p, q) \leq d_k(p)\}$.

2. Define the reachability distance of a point q from p, as $reach{-}dist_k(p, q) = \max\{d_k(q), d(p, q)\}$.

3. The average reachability distance is

$$reach - dist_{avg}(p) = \frac{\sum_{q \in \mathcal{N}_k(p)} reach - dist_k(p,q)}{|\mathcal{N}_k(p)|}.$$

The local reachability density of a point is defined as the reciprocal of reachability distance

$$\ell_k(p) = [reach - dist_{avg}(p)]^{-1}.$$

4. Finally, this local reachability density is compared with the local reachability densities of all points in $\mathcal{N}_k(p)$, and the ratio is defined as LOF (local outlier factor):

$$\mathcal{L}_k(p) = [\frac{\sum_{o \in \mathcal{N}_k(p)} \frac{\ell_k(o)}{\ell_k(p)}}{|\mathcal{N}_k(p)|}].$$

5. The LOF of each point is calculated, and points are sorted in decreasing order of $\mathcal{L}_k(p)$. If the LOF values are 'large', the corresponding points are declared as outliers.

6. To account for k, the final decision is taken as follows: $\mathcal{L}_k(p)$ is calculated for selected values of k in a pre-specified range, max $\mathcal{L}_k(p)$ is retained, and a p with large LOF is declared an outlier.

LOF performs well in many application domains, but its effectiveness will diminish if the density of an outlier is close to densities of its neighbors. In Figure 2.2, data object $o1$ is isolated but its density is very close to densities of objects in $C1$ which makes LOF fail to detect this outlier.

2.1.2 COF (Connectivity-based Outlier Factor) approach

To solve the deficiency found in the LOF, Tang *et al.*[58] suggest a new method to calculate the density as described below. Define the distance between two non-empty sets P and Q

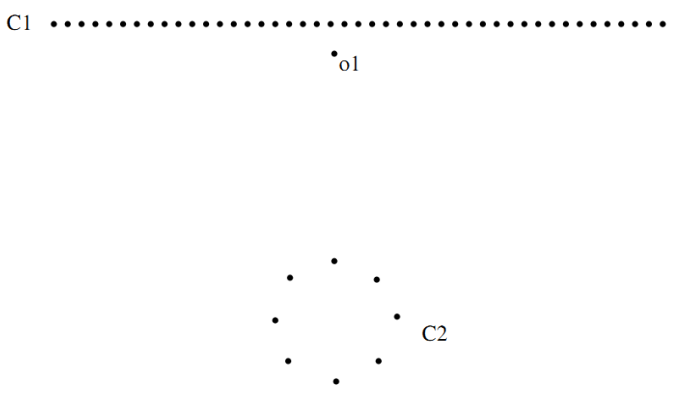

Fig. 2.2: **A case in which LOF fails for outlier detection.** - (This figure is from Tang *et al.*[58])

as $d(P,Q) = \min\{d(p,q) : p \in P, q \in Q\}$. This can be used to define the minimum distance between a point and a set by treating the point as a singleton set.

1. As in the previous algorithm, let $\mathcal{N}_k(p)$ be the set of k nearest neighbors of p.

2. Define set-based path (SBN) of length k as a path $< p_1, p_2, \ldots, p_k >$ based on the set $\{p, \mathcal{N}_k(p)\}$ such that for all $1 \leq i \leq k-1$, p_{i+1} is the nearest neighbor of the set $\{p_1, p_2, \ldots, p_i\}$ in $\{p_{i+1}, p_{i+2}, \ldots, p_k\}$. In other words, the SBN-path represents the order in which nearest neighbors of p are successively obtained. In simple words, SBN is an ordered list of all neighbors of p.

3. The Set-based trail (SBT) is an ordered collection of $k-1$ edges associated with a given SBN path $< p_1, p_2, \ldots, p_k >$. The ith edge e_i connects a point $o \in \{p_1, \ldots, p_i\}$ to p_{i+1} and is of minimum distance; i.e., length of e_i is equal to $d(o, p_{i+1}) = d(\{p_1, \ldots, p_i\}, \{p_{i+1}, \ldots, p_k\})$. Denote the length of edge as $dist(e_i)$. Figure 2.3 illustrates these concepts.

4. Given p, the associated SBN path $< p \equiv p_1, p_2, \ldots, p_k >$, and the SBT $< e_1, e_2, \ldots, e_{k-1} >$, the weight w_i for edge e_i is proportional to the order in which it is added to SBT set. Then the average-chaining distance ($\mathcal{A}$) of p is the weighted sum of the lengths of

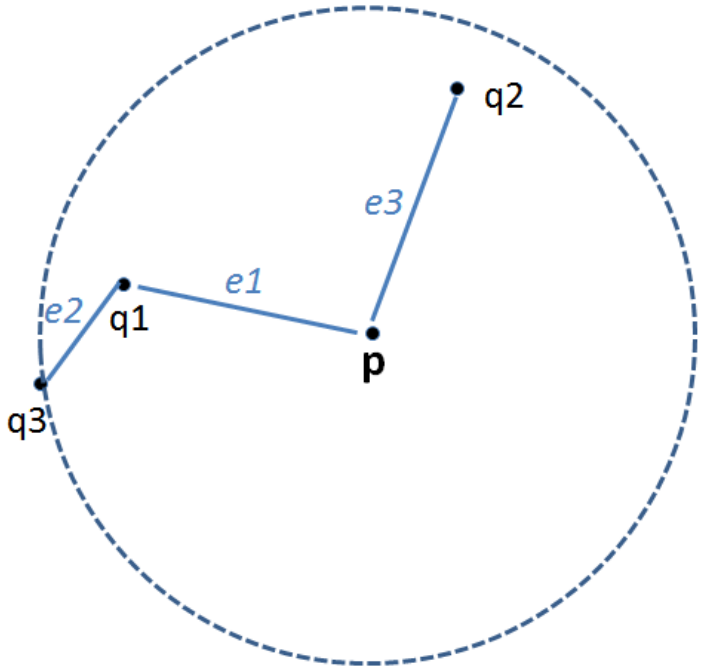

Fig. 2.3: **Illustration of SBN and SBT.** - The object p's k-nearest-neighbors are $q1, q2,$ and $q3$ when k = 3. Then SBN path from p is {p, q1, q3, q2}, and SBT is <$e1, e2, e3$> respectively.

the edges. That is:

$$\mathcal{A}_{\mathcal{N}_k(p)}(p) = \sum_{i=1}^{k-1} w_i \times dist(e_i).$$

where

$$w_i = \frac{2(k-i)}{k(k-1)}$$

5. Finally, the connectivity-based outlier factor (COF) of a point p is defined as the ratio of p's average-chaining distance with the average of average-chaining distances of its k nearest neighbors;

$$\text{COF}_k(p) = [\mathcal{A}_{\mathcal{N}_k(p)}(p)][\frac{\sum_{o \in \mathcal{N}_k(p)} \mathcal{A}_{\mathcal{N}_k(p)}(o)}{|\mathcal{N}_k(p)|}]^{-1}.$$

6. As in LOF, larger values of $\text{COF}_k(p)$ indicates that p is an outlier with higher possibility.

COF works better than LOF in the data sets with sparse neighborhoods (such as a straight line), but its computation cost is larger than LOF.

2.1.3 INFLO (INFLuential measure of Outlierness by symmetric relationship) approach

Proposed by Jin *et al.* [36], in INFLO the k nearest neighbors and reverse nearest neighbors of an object p are used to obtain a measure of outlierness.

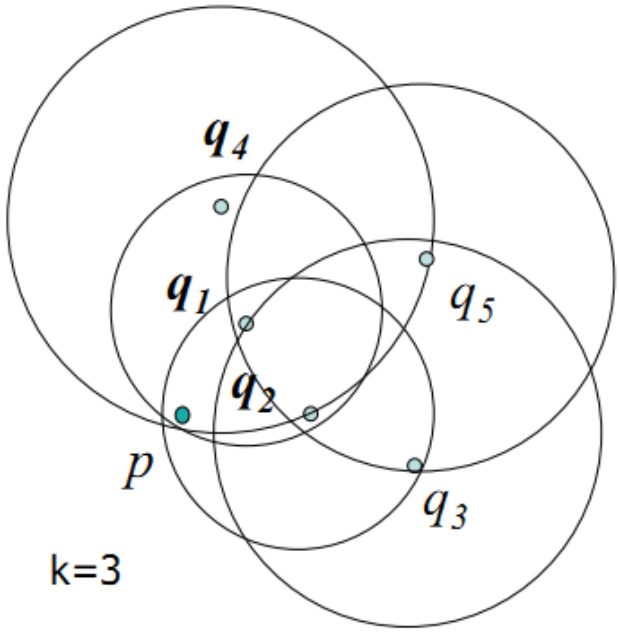

Fig. 2.4: **RNN and Influence Space** - For k=3, $\mathcal{N}_k(q_5)$ is $\{q_1, q_2, q_3\}$. $\mathcal{N}_k(q_1) = \{p, q_2, q_4\}$. $\mathcal{N}_k(q_2) = \{p, q_1, q_3\}$. $\mathcal{N}_k(q_3) = \{q_1, q_2, q_5\}$. $\mathcal{RN}_k(q_5) = \{q_3, q_4\}$. The original figure is from Jin *et al.* [36].

1. Reverse Nearest Neighborhood (RNN) of an object p is defined as

$$\mathcal{RN}_k(p) = \{q : \ q \in D \text{ and } p \in \mathcal{N}_k(q)\}.$$

Note that $\mathcal{N}_k(p)$ has exactly k objects but $\mathcal{RN}_k(p)$ may not have k objects. In some

instances, it may be empty, because for all $q \in \mathcal{N}_k(p)$, p may not be in any of the set of $\mathcal{N}_k(q)$.

2. The k-influential space for p, denoted as $\text{IS}_k(p) = \mathcal{N}_k(p) \cup \mathcal{RN}_k(p)$.

3. The influential outlierness of a point p is defined as

$$\text{INFLO}_k(p) = \frac{1}{\text{den}(p)} \frac{\sum_{o \in \text{IS}_k(p)} \text{den}(o)}{|(\text{IS}_k(p))|}$$

where $\text{den}(p) = \frac{1}{d_k(p)}$.

Thus for any p, INFLO expands $\mathcal{N}_k(p)$ to IS_k and compares p's density with average density of objects in IS_k. By using the reverse neighborhood, INFLO enhances its ability to identify the outliers in more complex situation, but it's performance is poor if an object p's neighborhood includes data objects from groups of different densities, then outlierness of p cannot be correctly measured.

2.2 Rank-Based Detection Algorithm (RBDA)

We present a new approach to identify outliers based on mutual closeness of a data point and its neighbors. The key idea of our algorithms is to use rank instead of distance. The idea of rank is borrowed from social network. Detecting outliers in a given data set resembles finding the most unpopular person in a given social network. Consider the social network in Figure 2.5: if we want to find if Bob is an unsocial person in it, then simply ask him this question:"who are your best friends?", then ask all "friends" of Bob the same question, if Bob is not in the list of his friends, he might be unpopular. Summaries of all answers are able to give a more clear conclusion about who unsocial person is. For example, in Figure 2.5 (a), Bob says that Eric is his closest friend but Eric says Bob is not one of his best friends. We ask the same question for every person in this social network. For example, the same questions for Eric and Jack show that they are best friends to each other. It can

be seen that Bob will appear to be No.1 unsocial person in this illustration. If we use the concept of ranks to capture the relationship of 'the best friend', then we can also use it to capture the outlierness of an outlier.

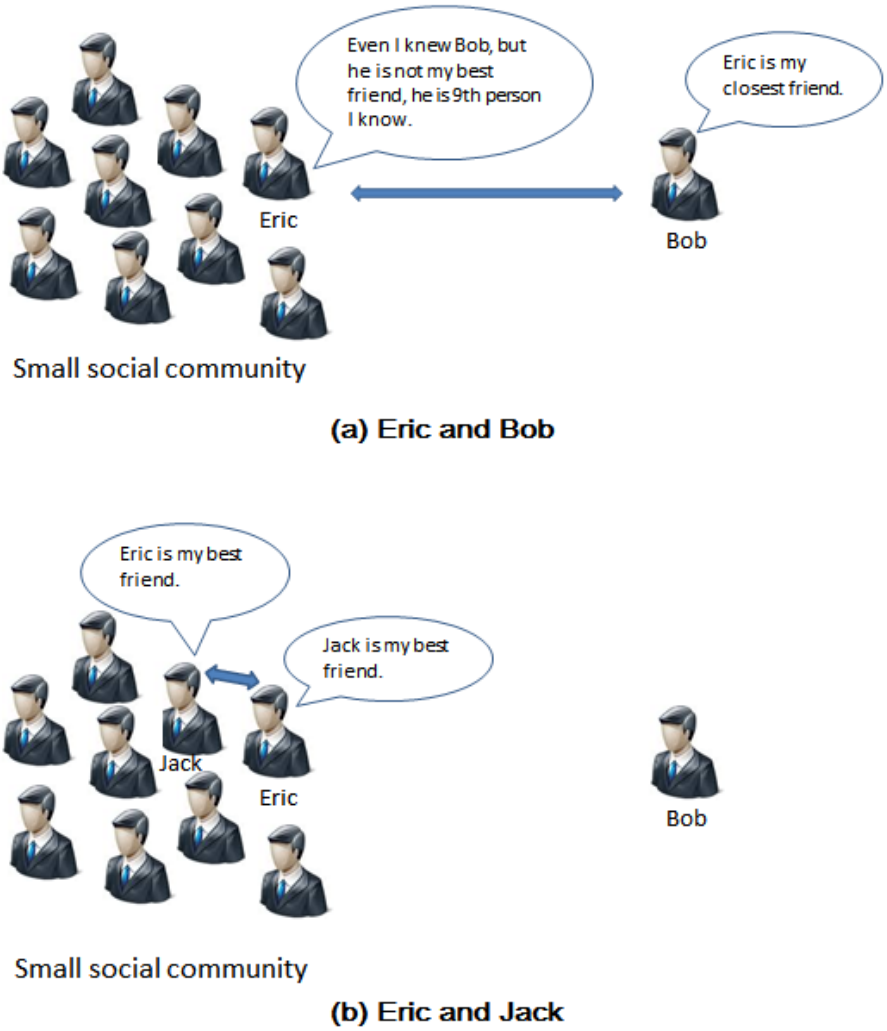

Fig. 2.5: **Rank in social network** - Rank can be used to measure popularity in a social network.

To understand mutual closeness consider $p \in D$ and $q \in \mathcal{N}_k(p)$. That is, consider a q which is "close" to p because it belongs to k-neighborhood of p. In return, we ask "how close is p to q?". If p and q are 'close' to each other, then we argue that (with respect to each other) p and q are not anomalous data points. However, if q is a neighbor of p but p is not a neighbor of q, then with respect to q, p is an anomalous point. If p is an anomalous point

with respect to most of its neighbors, then p should be declared to be an anomaly. When measuring the outlierness of p, instead of distance, we use the ranks calculated based on neighborhood relationships between p and $\mathcal{N}_k(p)$ in our outlier detection algorithms. This forms the basis of RBDA [31].

2.2.1 Description of Rank-based Detection Algorithm (RBDA) Algorithm

Algorithm: Rank-Based Detection Algorithm

Input: k, D

Output: List of RBDA values for each object p∈D

Method:

```
RBDAlist = NULL;                          /* Initialized output list. */
FOR each object p in D DO
    N(p) = NULL;                          /* Initialized p's neighborhood. */
    Nk(p) = NULL;
    FOR each object q in D DO
        IF q≠p THEN
            Add q to N(p);                /* Add q to p's D-neighborhood */
    Sort N(p) by dist(p, q) in ascending order;
    dk(p) = dist(p, qk);                  /* qk is the kth in N(p); */
    tmp = 0; index = 0; rank = 0;         /* Prepare for assigning rankings */
    FOR each object q in N(p) DO          /* From the closest neighbor to last neighbor */
        IF dist(p,q) ≤ dk(p) THEN
            Add q to Nk(p);               /* Add q to p's k-neighborhood. */
        index++;
        IF dist(p,q) ≠ tmp THEN
            rank = index;
        rp(q) = rank; tmp = dist(p,q);    /* Assign rankings */
FOR each object p in D DO
    sumranks = 0;
    FOR each object q in Nk(p) DO
        sumranks += rq(p) ;               /* Sum up all rankings with respect to neighbors */
    RBDAlist(p) = sumranks / |Nk(p)|;     /* Get p's average rankings */
```

Fig. 2.6: **Rank-based Detection Algorithm** - Pseudo code

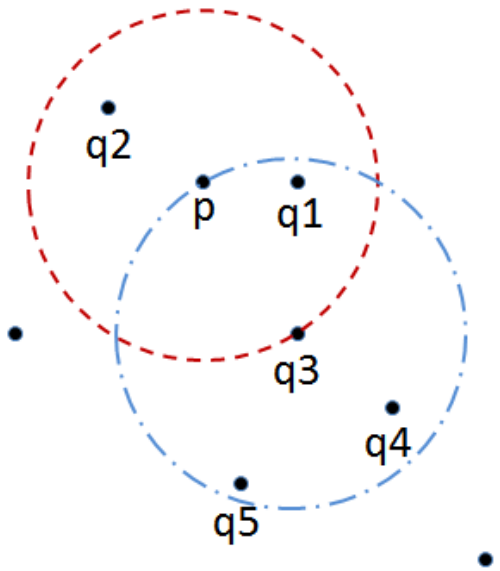

Fig. 2.7: **Illustration of ranks** - Red dash shows kNN of p when k is 3 and long blue dash shows a circle with radius of $d(p, q3)$ and center of $q3$. The kNN of p is {q1,q2, q3}. Then $r_{q3}(p)$=4, because $d(q3, p)$ is greater than any of $d(q3, q3), d(q3, q4), d(q3, q1)$ and $d(q3, q5)$.

1. Let $p \in D$ where $|D| = n$, $\mathcal{N}_k(p)$ denotes the set of its k-neighbors, $\mathcal{N}(p)$, denotes all the neighbors of $p \in D$, it is defined as $\mathcal{N}(p) = \{d(p, o); p \neq o, o \in D\}$. Let $r_q(p)$ denote the rank of p with respect to q, and it is defined as $r_q(p)$=$|I|$ where $I = \{d(q, o); d(q, o) < d(p, q), d(q, o) \in \mathcal{N}(q)\}$. For all $q \in \mathcal{N}_k(p)$, calculate the $r_q(p)$.

2. 'Outlierness' of p, denoted by $\mathcal{O}_k(p)$, is defined as:

$$\mathcal{O}_k(p) = \frac{\sum_{q \in \mathcal{N}_k(p)} r_q(p)}{|\mathcal{N}_k(p)|}. \tag{2.1}$$

   If $\mathcal{O}_k(p)$ is 'large' then p is considered an outlier.

3. One criterion to determine 'largeness' is described below. Let D_o = $\{p \in D \mid \mathcal{O}_k(p) \leq \mathcal{O}^*\}$ where $\mathcal{O}^*$ is chosen such that the size of D_o is a large fraction (e.g. 75%) of the

size of D. We normalize $\mathcal{O}_k(p)$ as follows:

$$\mathcal{L}_k(p) = ln(\mathcal{O}_k(p)) \tag{2.2}$$

$$Z_k(p) = \frac{1}{S_k}(\mathcal{L}_k(p) - \bar{\mathcal{L}}_k) \tag{2.3}$$

where

$$\bar{\mathcal{L}}_k = \frac{1}{|D_o|}\sum_{p\in D}\mathcal{L}_k(p) \text{ and } S_k^2 = \frac{1}{|D_o|-1}\sum_{p\in D}(\mathcal{L}_k(p) - \bar{\mathcal{L}}_k)^2$$

and if the normalized value $Z_k(p)$ is ≥ 2.5, then we declare that p is an outlier. In this criterion we have assumed that the distribution of $Z_k(p) = \frac{1}{S_k}(\mathcal{L}_k(p) - \bar{\mathcal{L}}_k)$ (normalized for mean and standard deviation) will be approximated by the standard normal random variable and $\mathrm{P}(Z_k(p) = \frac{1}{S_k}(\mathcal{L}_k(p) - \bar{\mathcal{L}}_k) > 2.5) \approx 0.006$. Hence, value of $Z_k(p) = \frac{1}{S_k}(\mathcal{L}_k(p) - \bar{\mathcal{L}}_k) > 2.5$ will be an outlier.

2.2.2 Why does RBDA work?

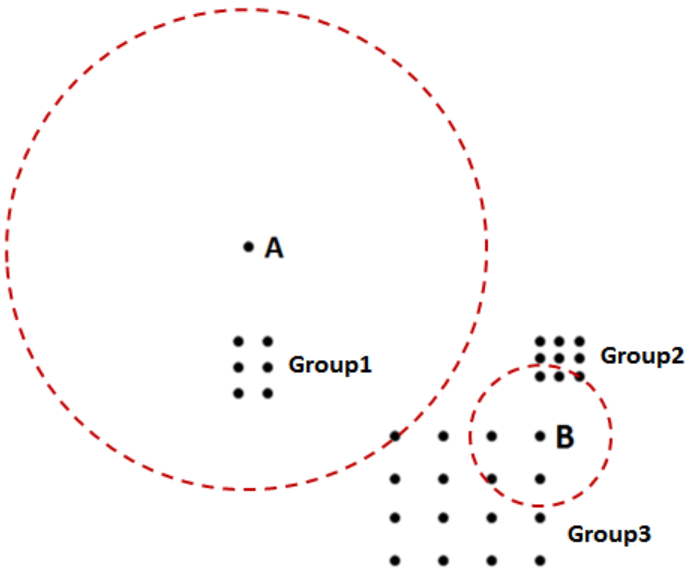

Fig. 2.8: **Illustration of a case that density-based algorithm would not work.** - Red dash circles contain the k nearest neighborhoods of 'A' and 'B' when k=7.

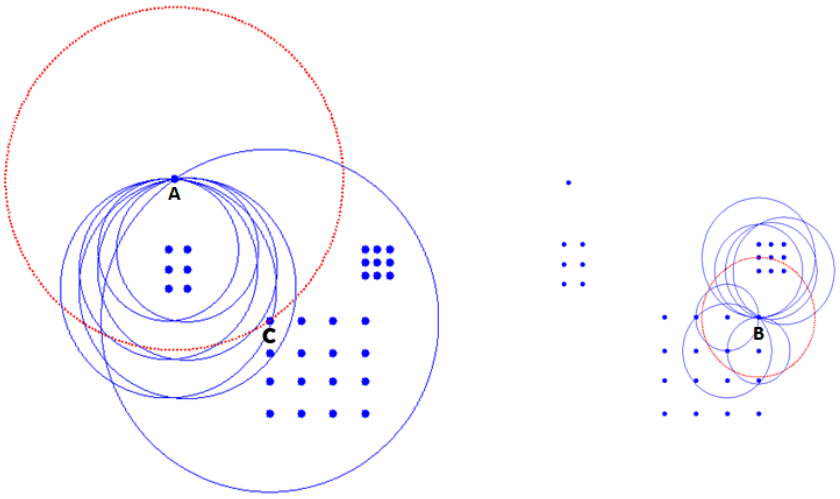

Fig. 2.9: **How rank works.** - Red dotted circle represents the k nearest neighborhood of 'A' when k is 7; Blue solid circle shows the range of I when calculating ranks of 'A' for each its kNN. The number of data objects in blue circle represents the rank of 'A' with respect to the object in the center of the circle. (Left) Ranks for 'A'; (Right) Ranks for 'B'

Before explaining why RBDA works, first we examine a scenario in which the density-based algorithm would fail. Consider the data set in Figure 2.8. There are three groups of data objects and one isolated data object - 'A'. Data object 'B' is from group3. When k is 7, the k nearest neighbors of both 'A' and 'B' contain the data objects from different density groups. In this case, the density-based outlier detection algorithm, LOF, assigns a higher outlierness value 1.5122 to 'B' and lower outlierness value 1.1477 to 'A' which is counter intuition. Density-based algorithms assume that all the neighbors of interested data objects are from the same density groups, but such is not the case in our example. In this illustration, LOF also suffers the same deficiency when k is 6, 7, or 8.

To overcome this issue, instead of focusing on the calculation of density, RBDA chooses another approach: using ranks instead of distance. Although rank is calculated based on distance, it contains more useful information. The basic idea is that if an object is an outlier, its rank with respect to its kNN is larger, whereas a normal object's rank is smaller.

Since RBDA requires calculating the ranks of the interested data object with respect to its k neighborhood, an object far away from many data objects in the data set will have

large rank. For example, in Figure 2.9, object 'C' is the 7th nearest neighbor of object 'A', but 'A' is the 31st neighbor of 'C', because 'C' is more close to the center of data set than 'A'. Thus in Figure 2.9, RBDA outlierness (average rank) of 'A' is 10 and that of 'B' is less (6.5714) as expected.

2.3 Experiments:

We use two synthetic and three real datasets to compare the performance of RBDA with LOF, COF and INFLO. Metrics to compare the algorithms are described below.

2.3.1 Metrics for Measurement

To evaluate the performance of the algorithms, three metrics were selected – precision, recall, and Rank-Power [2, 49, 57, 7].

Suppose the set D of n objects contains d_t true outliers. Suppose, using a given outlier detection algorithm, we identify m most suspicious instances in D. Let m_t be the number of true outliers among m instances. Then ***Precision***, which measures the proportion of true outliers in top m suspicious instances, is:

$$\text{Pr} = \frac{m_t}{m},$$

and *Re* which measure the accuracy of an algorithm is:

$$\text{Re} = \frac{|m_t|}{|d_t|}.$$

Precision and recall are insufficient to capture complete effectiveness of an algorithm. One algorithm may identify an outlier as the most suspicious while another algorithm may identify it as the least suspicious. Yet the values for the above two measures remain the same. Ideally, an algorithm will be considered more effective if the true outliers occupy top po-

sitions and non-outliers are among the least suspicious instances. “RankPower" was proposed by Tang *et al*. [58] to capture this notion. Let R_i denote the rank of the ith *true* outlier. Then,

$$\text{RP} = \frac{m_t(m_t+1)}{2\sum_{i=1}^{m_t} R_i}.$$

Rank-Power takes maximum value 1 when all n true outliers are in top n positions.

For a fixed value of m, larger values of all three of these metrics imply better performance.

2.3.2 Synthetic Datasets

Two synthetic datasets, shown in Figures 2.10 and 2.11, are used to evaluate the outlier detection algorithms. For illustrative convenience, we use only 2-dimensional data points so that outliers can be seen easily. In each dataset, there are multiple clusters with different densities. In each dataset, we have placed six additional objects, (A, B, C, D, E, and F) in the vicinities of the clusters to evaluate their ‘outlierness’ by LOF, COF, INFLO, and our proposed algorithm, RBDA.

In tables below, we summarize the performance of the algorithms, Pr represents precision, Re represents recall, and RP represents Rank-Power.

2.3.2.1 Synthetic Dataset 1

Synthetic dataset 1 contains four clusters of different densities consisting of 36, 8, 8, and 16 instances. Four different values of k and four values of m are used; results are shown in Table A.1 in Appendix.

For $k = 4$, only COF and RBDA algorithms find all six outliers within top m ranked instances, and their RankPowers are higher than those of LOF and INFLO algorithms.

For $k = 5$, 6, and 7 RBDA has higher precision and recall than the other algorithms; and RBDA is the only algorithm that obtains the maximum RankPower of 1 for every m.

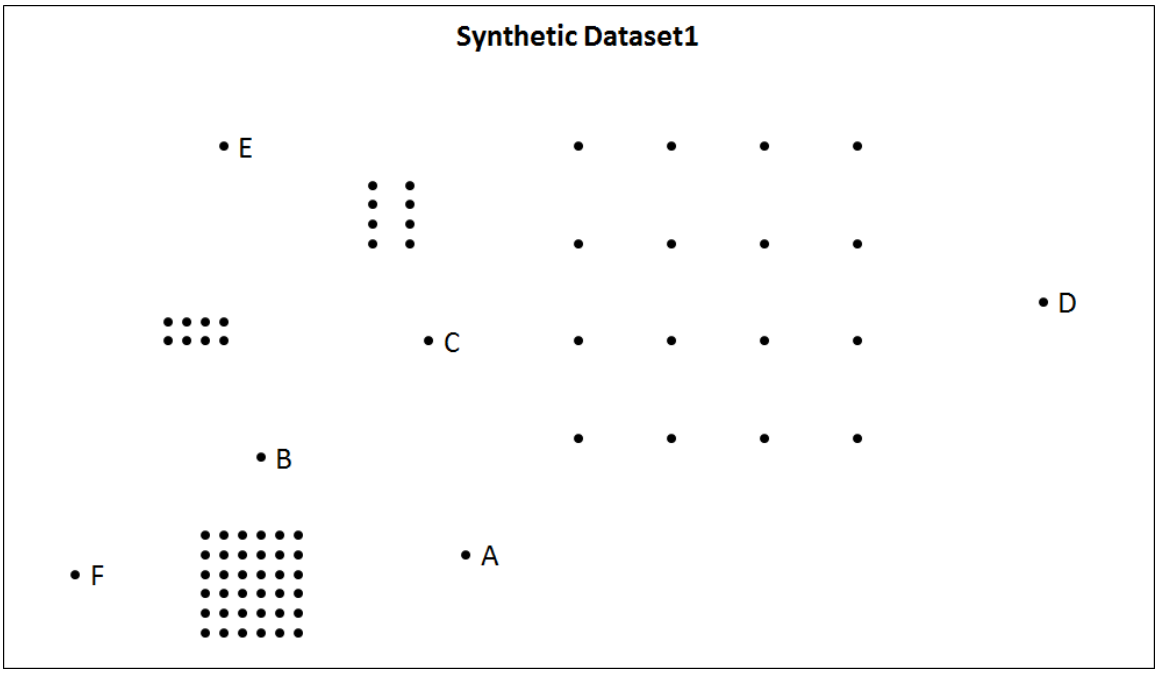

Fig. 2.10: **A Synthetic dataset with clusters obtained by placing all points uniformly with varying degrees of densities.** -

2.3.2.2 Synthetic Data set 2

Synthetic dataset 2 consists of 515 data objects including six planted outliers; this data set has one large normally-distributed cluster and two small uniform clusters.

Results are presented in Table A.2 in Appendix for $k = 25, 35,$ and 50, and $m = 5, 10, 15, 20$ and 30. It can be seen that for $k = 25$ and 35, RBDA has the best precision, recall and Rank-Power for all m from 5 to 30. When k is increased to 50, RBDA still performs better than others except for m = 20. The reason why LOF doesn't work for k=50 is shown in Figure 2.12: the true outlier o has shorter distance to its kth nearest neighbor than object q. When comparing the ratios of densities, q's value is larger than o so that it makes q look more like an outlier than o.

In this experiment, RBDA algorithm works better than others in most of the cases, and for k = 25 and 35, it achieves the best performance.

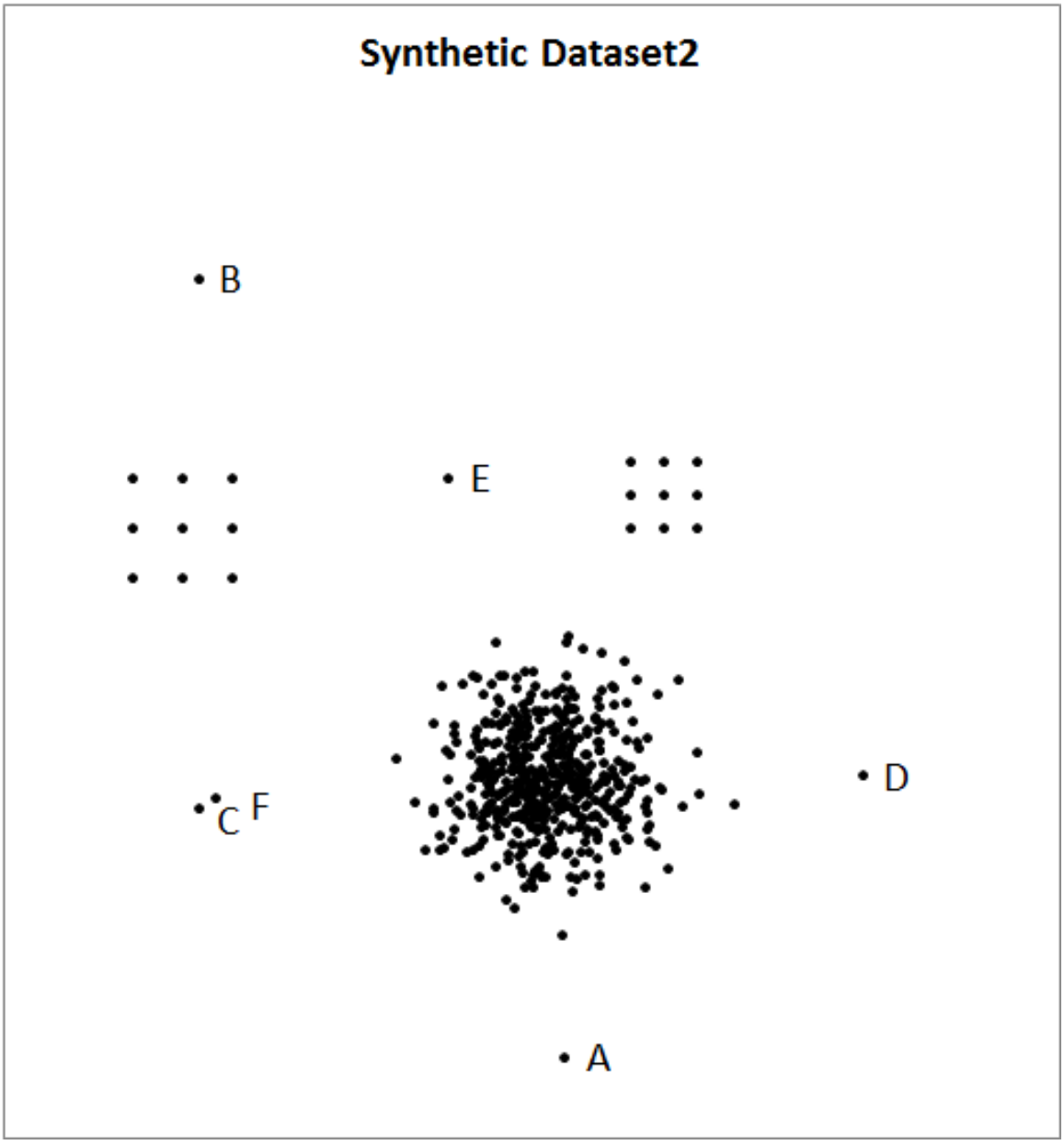

Fig. 2.11: **Synthetic data set 2** - A synthetic data set with one cluster obtained using the Gaussian distribution and other clusters by placing points uniformly.

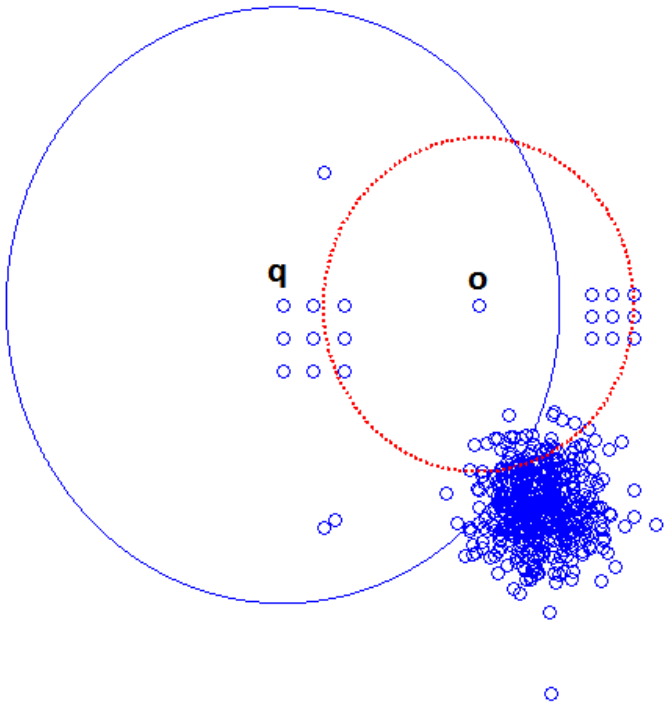

Fig. 2.12: **Why LOF does not work.** - Red dash circle encloses the k nearest neighbors (kNN) of object o which is an outlier, and blue circle represents kNN of object q when k is 50.

2.3.3 Real Datasets:

We have used three well known datasets, namely the Iris, Ionosphere, and Wisconsin breast cancer datasets. We use two ways to evaluate the effectiveness and accuracy of outlier detection algorithms; (i) detect rare classes within the datasets, rare classes are generated by removing a majority of objects from the original class (this methodology has also been used by other researchers such as Feng *et al.* and Tang *et al.* [22, 58]) and (ii) plant outliers into the real datasets (according to problem specific knowledge) and expect outlier detection algorithms to identify them.

2.3.4 Real Datasets with Rare Classes

In this subsection, we compare the algorithms in detecting rare classes. A class is made 'rare' by removing most of its observations. In all cases, the value of k is chosen between 5% to 10% percentage of the size of the dataset. Because the attributes are dependent, Mahalanobis distance is used to measure the distance between two points.

2.3.4.1 Iris Dataset

This well-known data set contains the categorization of iris flowers to three classes: Iris Setosa, Iris Versicolour, Iris Virginica, with 50 instances each. The Iris Setosa class is linearly separable from the other two classes, but the other two classes are not linearly separable from each other. We randomly remove 45 instances from Iris Setosa class to make it 'rare'; the remaining 105 instances are used in the final dataset. Three selected values of k are 5, 7, 10. In Appendix, Table A.5 summarizes our findings.

We observe that for $k = 5$, LOF and RBDA both have the highest precision and recall values while RBDA has the best RankPower for all values of m; COF performs poorly (its precision and recall values are all zero). The reason for COF's poor performance is that instances of rare class are close to each other which decrease average-chaining distance of

COF algorithm significantly and thus decrease its usefulness.

For $k = 7$, LOF performs better than RBDA. In particular, COF doesn't find any outlier within top 10 ranked instances.

For $k = 10$ and all values of m, LOF, INFLO and RBDA perform well.

2.3.4.2 Johns Hopkins University Ionosphere Dataset

The Johns Hopkins University Ionosphere dataset contains 351 data objects with 34 attributes; all attributes are normalized in the range of 0 and 1. There are two classes labeled as good and bad with 225 and 126 data objects respectively. There are no duplicate data objects in the dataset. To form the rare class, 116 data objects from the bad class are randomly removed. The final dataset has only 235 data objects with 225 good and 10 bad data objects. Four values of k = 11, 15, 20 and 23 are used; values of m are assigned at 5, 15, 30, 60 and 85. Results are presented in Appendix Table A.3.

We observe that, for k = 11, among all algorithms RBDA has the best precision, recall and RankPower except for m = 15. LOF and INFLO algorithms achieve the best RankPower for m = 15; but RBDA is the only algorithm that finds all ten 'bad' class instances.

For k = 15, LOF has higher RankPower than RBDA for m = 15 and 30; but for other values of m, RBDA is the winner and has largest values of RankPower. RBDA algorithm finds all 10 'bad' class instances while other algorithms only find 9 of them when m is less than 85. For k = 20 and 23, situation is very similar to the previous case.

In general, RBDA shows the best overall performance.

2.3.4.3 Wisconsin Diagnostic Breast Cancer Dataset

Wisconsin diagnostic breast cancer dataset contains 699 instances with 9 attributes. There are many duplicate instances and instances with missing attribute values. After removing all duplicate instances and instances with missing attribute values, 236 instances labeled

as benign class and 236 instances as malignant were left. Following the method proposed by Cao [7], 226 malignant instances are randomly removed. In our experiments the final dataset consisted 213 benign instances and 10 malignant instances. Results are presented in Appendix Table A.4.

For k = 11, RBDA achieves the best precision and recall for m = 15, 20 and 50, and gets the best RankPower for m = 30 and 40. COF has the best precision and recall for m = 30, 40 and 50, and it also has the best RankPower for m = 15, 20 and 50.

For k = 15, 19 and 22, the relative performance results are similar. Either RBDA or COF gets the best precision, recall and RankPower for every m. But none of them can achieve the best performance for all values of m.

In general, COF performs a little better than RBDA, but for certain values of m, RBDA has better RankPower than COF.

2.3.5 Real Datasets with Planted Outliers

In experiments described in this subsection we plant some outliers into the real datasets according to datasets' domain knowledge.

2.3.5.1 IRIS with Outliers

We insert three outliers into IRIS dataset, that is, there are three classes with 50 instances each and 3 planted outliers. The first outlier has maximum attribute values, second outlier has minimum attribute values, and the third has two attributes with maximum values and the other two with minimum values. Appendix Table A.6 contains the results for this setting.

For $k = 7$ and $m \geq 10$, all algorithms found the three outliers, but their RankPowers are different. RankPower of COF is lower than those of the other three algorithms; LOF, INFLO and RBDA all have the best performance for precision, recall and RankPower.

For k = 15, INFLO and RBDA are the best because they all rank three outliers in top three positions which are exactly the expected results that outlier detection algorithms are

designed to do. LOF and COF have the worst Rank-Power.

2.3.5.2 *Johns Hopkins University Ionosphere Dataset with Outliers*

For ionosphere dataset, two classes labeled as good and bad, with 225 and 126 instances respectively, are kept in the resulting dataset. Three outliers are inserted into the dataset; the first two outliers have maximum or minimum value in every attribute, and the third has 9 attributes with unexpected values and 25 attributes with maximum or minimum values. The unexpected value here is a value that is valid (between minimum and maximum) but is never observed in real datasets[1].

In Appendix Table A.7, we observe that RBDA consistently performs better than the other algorithms. For all values of k and m, RBDA achieves the best precision, recall and RankPower. In addition, RBDA algorithm is the only algorithm that detects all three rare class instances for all three k values when m is 40.

The gap of performance between RBDA and other algorithms is large since RBDA's RankPower is almost twice that of other algorithms, for every k and m. The overall performance of RBDA is the best.

2.3.5.3 *Wisconsin Diagnostic Breast Cancer with Outliers*

After removal of duplicate instances and instances with missing attribute values, only 449 instances were left with 213 instances labeled as benign and 236 as malignant. Two outliers were planted into dataset. Both outliers have maximum or minimum values for all attributes.

In Appendix Table A.8, for every k, our algorithm, RBDA, is the only algorithm that has the maximum RankPower (1) for every value of m. It also has the best precision and recall for all k and m. We can observe that even the maximum RankPower of LOF, COF

[1]For example, one attribute may have a range from 0 to 100, but the value of 12 never appears in real dataset.

and INFLO is only half of the RankPower of RBDA, which means that their success in outlier detection is much less than that of RBDA in this experiment.

In general, when k increases, most of the algorithms improve their performances. One reason is that when k is larger, more neighbors around a specific instance are involved in the process of evaluating that instance, so that the algorithm holds more information to make an accurate decision about outliers.

2.3.6 Conclusions

Outlier detection is an important task for data mining applications. Existing algorithms are effective and have been successfully applied in many real-world applications. But these algorithms, especially density-based algorithms, have low efficacy in datasets with different densities. In our work, we have introduced the Rank-based outlier Detection Algorithm (RBDA) that is effective in solving the problem mentioned above.

CHAPTER 3
ANOMALY DETECTION ALGORITHMS BASED ON CLUSTERING AND WEIGHTED RANKS

Clustering can be used to find anomalous objects in a given data set. This approach has a significant advantage; it reduces the time complexity considerably provided the clustering algorithm is fast.

In this chapter we argue that an object that belongs to a very large cluster is more likely to be declared anomalous even if it is only slightly far away from its neighbors, compared to the case if it had belonged to a small cluster. To account for this imbalance, we propose the concept of modified rank.

The chapter is organized as follows. In Section 3.1 we discuss some clustering approaches for anomaly detection. In Section 3.2 we introduce important notations, and in Section 3.3 we discuss our clustering algorithms, followed by our proposed new algorithm in Section 3.4. In Section 3.5, we compare the performance of the new algorithm with RBDA and other close competitors.

3.1 Clustering Approach for Anomaly Detection and a New Clustering Algorithm

Among existing anomaly detection algorithms, one unsupervised approach is based on clustering using the assumption that normal data objects belong to a cluster whereas anomalies do not belong to any clusters. Its essential theme is to find clusters in a given data set and declare any data object that does not belong to any cluster as anomalous. Unlike the classical approaches to clustering in which all objects of the data set must belong to a cluster, in this context objects are not required to be in a cluster. K-mean clustering and other well known clustering techniques cannot be applied towards this task; special clustering algorithms have been developed that do not force every data object to belong to a cluster. DBSCAN, ROCK and SNN clustering algorithms proposed by Ester *et al.* [18], Guha *et al.*, [28], and Ertoz *et al.* [17], respectively, are among such clustering algorithms.

Unlike previous work, we do not require that all objects that do not belong to a cluster are anomalous; instead we assume that data objects that do not belong to a cluster are 'potential' anomaly candidates. Further investigation is required to evaluate their anomalousness.

In Section 3.3 we describe a new clustering algorithm which exploits "reachability". The main idea is to declare data objects p and q to be 'close' if q is one of the k nearest neighbor of p and vice versa. We define "reachability" as the transitive closure of the 'closeness' relation, e.g; if $p1$ and $p2$ are 'close' and $p2$ and $p3$ are 'close' we declare that $p3$ is reachable from $p1$. Finally, reachable data objects are consider to form a cluster. Details in mathematical notation are presented below.

There is one lingering concern related to the size of a cluster. For instance, if a cluster contains only two data objects, is it 'really a cluster'? If anomalies in the data set form clusters by themselves, then a cluster based approach won't be able to detect them, leading

to poor performance. Chandola *et al.* [11] address such concerns. A subjective judgment has to be made regarding the minimum number of data objects to form a cluster recognizing yet another assumption to detect anomalies: “Normal data objects belong to large and dense clusters, while anomalies either belong to small or sparse clusters."

We proposed a new algorithm by using cluster size and distance (in addition to the rank) directly in evaluating the outlierness of data objects. We present several variations of such algorithms, all of which begin with clustering as a preprocessing step. These algorithms are evaluated on real and simulated test data sets, helping to understand the kinds of problems in which they are most useful. These variations exhibited similar performance to each other, which was considerably better than that of algorithms explored in previous work.

In the next section, after introducing key notations and definitions, we describe Tao and Pi’s [60] DBCOD algorithm. Next we present new outlierness measures and new algorithms for outlier detection. These new algorithms are compared with RBDA, DBCOD, and other algorithms, using one synthetic and three real data sets. Brief descriptions of data sets are presented in Section 3.5, followed by our findings. Section 3.6 presents concluding remarks.

3.2 Notation and Definitions

The following definitions are used in the proposed clustering algorithm; all definitions are parameterized with a positive integer parameter ℓ intended to capture the notion of cluster tightness.

- *D-reachability* (given ℓ): An object p is directly reachable (D-reachable) from q, if $p \in \mathcal{N}_\ell(q)$.

- *Reachability*: An object p is reachable from q, if there is a chain of objects $p \equiv p_1, \ldots, p_n \equiv q$, such that p_i is D-reachable from p_{i+1} for all values of i.

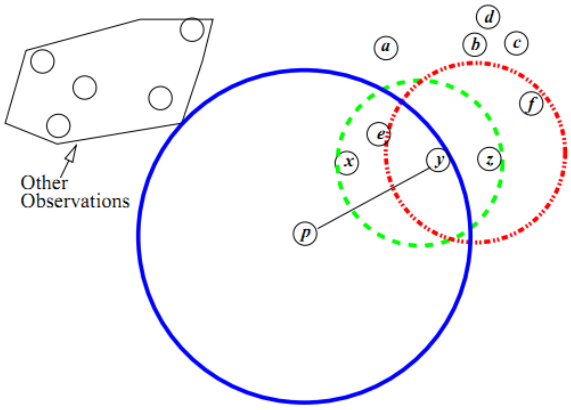

Fig. 3.1: **Illustrations of $d_k(p)$, $\mathcal{N}_k(p)$ and $\mathcal{RN}_k(p)$ for $k = 3$.** - The large blue circle contains elements of $\mathcal{N}_3(p) = \{x, e, y\}$. Because y is farthest third nearest neighbor of p, therefore $d_3(p) = d(p, y)$, the distance between p and y. The smallest (green) circle contains elements of $\mathcal{N}_3(y)$ and the medium radius circle (red) contains elements of $\mathcal{N}_3(z)$. Note that $r_y(p) =$ (rank of p among neighbors of y) = 9. Finally, $\mathcal{RN}_3(p) = \varnothing$, since no other object considers p in their neighborhood.

- *Connectedness*: If p is reachable from q, and q is reachable from p, then p and q are connected.

Figure 3.1 illustrates the concepts of $d_k(p)$, $\mathcal{N}_k(p)$ and $\mathcal{RN}_k(p)$.

3.3 Neighborhood Clustering (NC-clustering)

We use breadth-first search on a graph whose node-set is D and in which an edge exists between $p, q \in D$ if $p \in \mathcal{N}_\ell(q)$ and $q \in \mathcal{N}_\ell(p)$. A connected component C of the graph is a cluster if the following conditions are satisfied:

1. Every object in C is D-reachable from at least two others in C.

2. The number of objects in C is no smaller than a pre-specified minimum, m^*.

We denote the clustering method as $\mathcal{NC}(\ell, m^*)$. For example, $\mathcal{NC}(6, 5)$ denotes that a cluster contains connected objects for ℓ = 6, and every cluster must contain at least 5 objects.

If any connected component C does not satisfy these conditions, it is broken up into isolated objects, which are declared to be potential outliers.

In Figure 3.1 for $k = \ell = 3$, y is D-reachable from p since y is in $\mathcal{N}_3(p)$. but p and y are not connected since p is not in $\mathcal{N}_3(y)$. However, y and z are connected since they are in each other's 3-neighborhoods.

The appropriate values of ℓ and m^* are problem-specific and depend on domain knowledge. If ℓ is small, the NC-clustering method will find small and tightly connected clusters. On the other hand, large values of ℓ will result in large and loose clusters. If the clusters are small and tight, we expect to find more objects that don't belong to any cluster whereas in the latter case, only a few objects will be declared as outliers. In real world applications (such as credit card fraud detection) most of the transactions are normal and only 0.01% or less of the transactions are fraudulent. In such cases, a small value of ℓ is more suitable than a large ℓ.

The value of m^* has a similar effect: if m^* is too small, then the cluster size may also be too small, and a small collection of outliers may be considered as a cluster, which is not what we want. In our experiments, m^* is set to a fixed value of 6.

Advantages of $\mathcal{NC}(\ell, m^*)$ clustering algorithms are:

- It only requires one scan to find all the clusters.
- It controls the tightness and sparsity of clusters using a single parameter - ℓ.
- It can find the central objects of clusters easily by analyzing each $\mathcal{O}_k(p)$. The data objects in the center must have lower $\mathcal{O}_k(p)$ values.

These properties will be further investigated in future work but are not fully evaluated in this thesis, since the focus of this dissertation is not on developing or evaluating clustering

algorithms.

3.3.1 Density and Rank Based Detection Algorithms

More recently, a density-based clustering and outlier detection algorithm (DBCOD) has been proposed by Tao and Pi [60], described below

Density-based clustering and outlier detection algorithm (DBCOD) For $p \in D$, Tao and Pi [60] define the local density, the neighborhood-based density factor, and neighborhood-based local density factor of p, respectively, as:

$$\text{LD}_k(p) = \frac{\sum_{q \in \mathcal{N}_k(p)} \frac{1}{d(p,q)}}{|\mathcal{N}_k(p)|}, \text{ NDF}_k(p) = \frac{|\mathcal{RN}_k(p)|}{|\mathcal{N}_k(p)|}, \text{ and NLDF}_k(p) = \text{LD}_k(p) \times \text{NDF}_k(p).$$

The threshold of NLDF, denoted as τ_{NLDF}, is defined as:

$$\tau_{\text{NLDF}} = \begin{cases} \min_k(\text{NLDF}_k(p)) & \text{if for all objects } p \in D, \text{NDF}_k(p) = 1 \\ \max_k(\text{NLDF}_k(p)) & \text{otherwise} \end{cases}$$

Using the above definitions, Tao and Pi [60] find the clusters based on the definitions in Section 3.2, *except their definition of D-reachability is as follows:* p *and* q *are in each other's* k*-neighborhood and NLDF*$_k(q) < \tau_{NLDF}$. Points outside the clusters are declared as outliers.

3.4 New algorithms based on distance and cluster density

Purely rank-based analysis leads to potential incorrect answers when an object is near a dense cluster; this property is the 'cluster density effect'. For instance, two points are of special interest in Figure 3.2: point 'A' in the neighborhood of a cluster with low density (25 objects) and point 'B' in the neighborhood of a cluster with high density (491 objects).

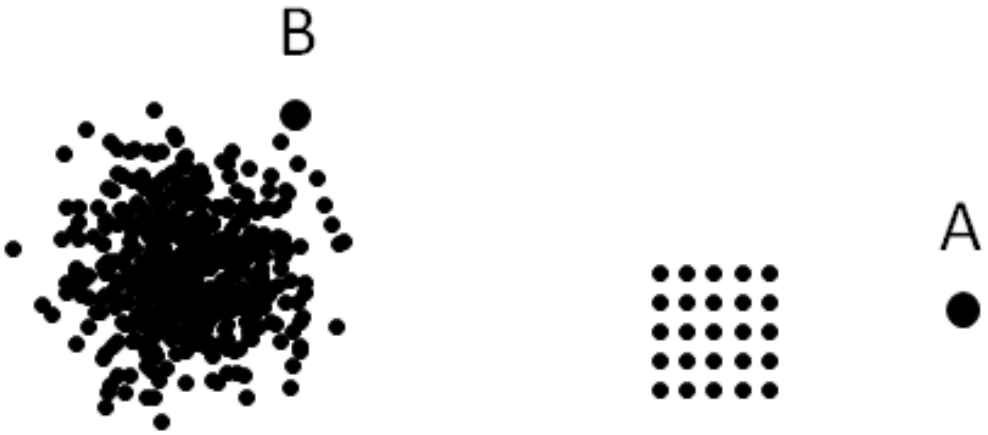

Fig. 3.2: **An example to illustrate 'Cluster Density Effect' on RBDA; RBDA assigns larger outlierness measure to B.** -

By visual inspection, it would be argued that the object 'A' is an outlier whereas object 'B' is a possible but not definite outlier. For k=20, $O_{20}(A)$=25 because rank of 'A' is 25 from all of its neighbors. On the other hand, the ranks of 'B' with respect to its neighbors are: 2, 8,..., 132, 205, 227; so that $O_{20}(B)$ is 93.1. RBDA concludes that 'B' is more likely to be outlier than 'A'. This is due to the presence of a large and dense cluster in the neighborhood of 'B', hence a point close to a dense cluster is likely to be misidentified as an outlier.

By visual inspection, we intuitively conclude that a point is an outlier if it is 'far away' from the nearest cluster. This implies that the distance of the object (from the cluster)

plays an important role; but in RBDA the distance is accounted for indirectly, only through rank. This motivates examining possible improvements that may be obtained by modifying the outlierness measure to give additional weight to the distance of a point from the set containing its neighbors. In the first step, the data is clustered using a clustering algorithm, and in the second step, a data object's anomaly score is evaluated as its distance from the closest centroid. We use a modification of this approach. Note that we don't use the centroid of the clusters, mainly due to the reason that clusters generated by the schema discussed in Section 3.3 are not necessary spherical; we instead use the centroid of the point's k nearest neighbors. A distance could be defined in multiple ways, $e.g.$, three possible distance measures of a point q from the set $\mathcal{N}_k(p)$ are:

- $\min_{q \in \mathcal{N}_k(p)} d(p,q)$ (minimal distance)
- $\max_{q \in \mathcal{N}_k(p)} d(p,q)$ (maximal distance)
- $\frac{1}{|\mathcal{N}_k(p)|} \sum_{q \in \mathcal{N}_k(p)} d(p,q)$ (averaged distance)

Such different distance measures lead to multiple measures of outlierness. We have explored some of them but in the next subsection we consider only the third (averaged distance) that resulted in slightly better outlier detection performance than the other two variations.

3.4.1 Rank with Averaged Distance Algorithm (RADA)

This algorithm, described below, adjusts the rank-based outlierness value by the average distance of p from its $k-$neighbors, where k, ℓ, m^* are positive integer parameters.

1. Find the clusters in D by $\mathcal{NC}(\ell, m^*)$ method.
2. Declare an object o to be a *potential-outlier* if it does not belong to any cluster.

3. Calculate a measure of outlierness:

$$W_k(p) = \mathcal{O}_k(p) \times \frac{\sum_{q \in \mathcal{N}_k(p)} d(q,p)}{|\mathcal{N}_k(p)|} \tag{3.1}$$

where $\mathcal{O}_k(p)$ is as defined in Equation 2.1 in the description of RBDA.

4. If p is a potential-outlier and $W_k(p)$ exceeds a threshold, declare p to be an outlier.

For the dataset in Figure 3.2, we observe that $W_{20}(\mathrm{A}) = 484.82$ and $W_{20}(\mathrm{B}) = 396.19$ implying that A is more likely outlier than B, illustrating that RADA is capable of overcoming the problem observed with RBDA.

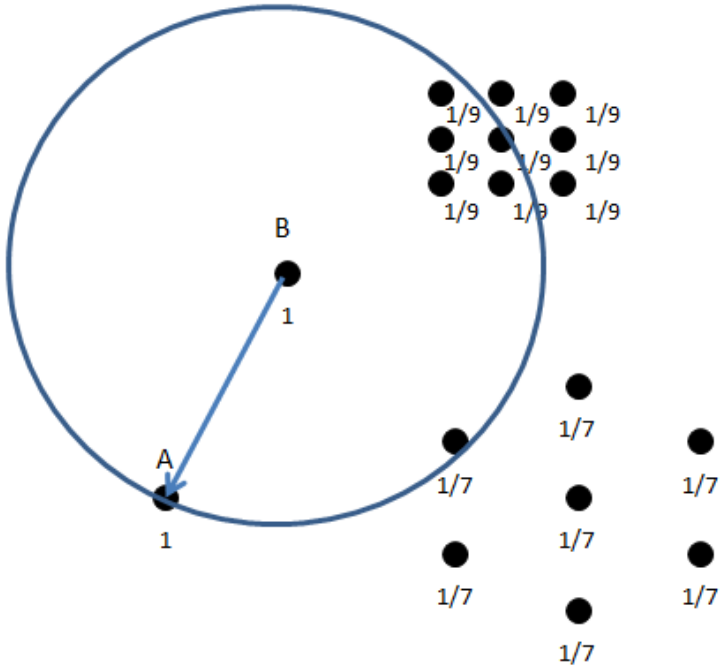

Fig. 3.3: **Assignment of weights in different clusters and modified-rank.** - Modified-rank of A, with respect to B, is $1 + 5 \times \frac{1}{9} + \frac{1}{7}$.

3.4.2 Outlier detection using modified-ranks (ODMR)

In this section we propose three alternative procedures to overcome the cluster density effect. We have observed that the size of the neighboring cluster plays an important role

when calculating the object's outlierness via RBDA. Recall that in RBDA the weight of a cluster C is $|C|$. But, in general, we can assign it a smaller weight to reduce the size effect. Weight assignments equal to 1, $\sqrt{|C|}$, and $\log|C|$ are some possible examples.

We have experimented with weight assignment = 1 and $\sqrt{|C|}$: associated statistics are described below.

3.4.2.1 ODMR:

Suppose that all clusters (including isolated points viewed as a cluster of size 1) are assigned weight 1, i.e., all $|C|$ observations of the cluster C are assigned equal weights = $1/|C|$.

Then the "modified-rank" of p is defined as $mr_q(p) =$ the sum of weights associated with all observations within the circle of radius $d(q,p)$ centered at q.

The desired statistic is the sum of "modified-ranks" in $q \in \mathcal{N}_k(p)$, denoted by

$$\Omega = \sum_{q \in \mathcal{N}_k(p)} mr_q(p). \tag{3.2}$$

Figure 3.3 illustrates how modified-rank is calculated in this case.

3.4.2.2 ODMRS:

If cluster C is assigned a weight = $\sqrt{|C|}$, i.e., each observation of the cluster is assigned the weight = $1/\sqrt{|C|}$, then the "modified-rank" of p is obtained by summing these weights associated with all observations within the circle of radius $d(q,p)$ centered at q; that is

$$\text{modified-rank of } p \text{ from } q = mr_q^S(p) = \sum_{s \in \{d(q,s) \le d(q,p)\}} \text{weight}(s).$$

The associated statistic $\Omega^{(S)} = \sum_{q \in \mathcal{N}_k(p)} mr_q^S(p)$.

3.4.2.3 ODMRW:

Yet, a third alternative, to define modified rank, is as follows. Given a cluster C, first we define $p_{k,d} = \sum_{q \in \mathcal{N}_k(p)} d(p.q)$. Then the modified rank of p is defined as:

$$mr_q^W(p) = \frac{p_{k,d}}{\sum_{p \in \mathcal{C}} p_{k,d}}.$$

The associated statistic is $\Omega^{(W)} = \sum_{q \in \mathcal{N}_k(p)} mr_q^W(p)$

3.4.2.4 ODMRD:

Influenced by the distance consideration of Section 3.4.1, we present one more algorithm that modifies ODMR by using additional distance information. ODMRD-outlierness, denoted as ${\Omega^{(D)}}_k(p)$, is defined as:

$${\Omega^{(D)}}_k(p) = \sum_{q \in \mathcal{N}_k(p)} mr_q(p) \times d(q,p)$$

Note that we have used the modified rank definition of ODMR. Alternatives such as ODMRS and ODMRW could also be explored.

3.4.3 Algorithm Description

Each of the above variations of ODMR can be used in an algorithm to compute outlierness. We use Ω in the description below, representing ODMR, whereas other variations use $\Omega^{(W)}, \Omega^{(S)}$, and $\Omega^{(D)}$, respectively, whose definitions were given above. As before, algorithm parameters k, ℓ, m^* are positive integers, and D is an non-empty data set. The algorithm is as follows:

1. Cluster the points in D using the $\mathcal{NC}(\ell, m^*)$ algorithm;

2. Let the set of potential outliers $P =$ the set of points in D that do not belong to any cluster;

3. For each point $p \in P$, compute $\Omega(p)$ (as defined in Equation 3.2 or its variants);

4. Points p with large values of $\Omega(p)$ are declared to be outliers.

When applied to a new problem for which the ground truth is not known, we may examine the distribution of Ω values to determine a threshold and all points whose Ω values exceed this threshold would be considered to be outliers. To find the threshold, one can use the classical upper 95% percentile of Ω values subject to the condition that there must be a drastic difference between the average of the first 95% values versus the last 5% values (e.g. 4 times more). When there is no such threshold, e.g., if Ω values are uniformly distributed in an interval, the user would have to select a threshold parameter m that indicates how many points are to be considered outliers. Two algorithms could be compared by examining the sets of m points considered to have the highest outlierness values (Ω), and evaluating in each case how many of these are true outliers (if ground truth is known).

In summary, we have proposed five modifications to RBDA: weighted RBDA, three based on modified ranks of p among its $k-$neighbors, and one with modified ranks in conjunction with the distance.

3.5 Experiments

In this section, we compare the performance of RADA, ODMR, ODMRS, ODMRW, ODMRD, RBDA, DBCOD, LOF, COF and INFLO for several datasets described in the experiment section of previous chapter.

3.5.1 Results

For the synthetic data set, results presented in the Apprendix, Tables B.1 and B.2, show that RADA, ODMR, ODMRS, ODMRW and ODMRD work extremely well for all values of k. RBDA achieves the best performance only when k is 25. But all the new algorithms work better than LOF, COF, INFLO and DBCOD.

For iris data set with rare class, in Tables B.3 and B.4 results show that ODMRW achieves the best performance when k is 5; LOF is the best when k is 10, while ODMRW is the second best. For k is 10, RADA, ODMRW, RBDA, LOF, and INFLO have the best performances.

For ionosphere data set with rare class, results in Tables B.5 and B.6 (with respect to metrics m_t and Recall) show that the algorithms RBDA, ODMR, ODMRD, and RADA perform equally well whereas DBCOD does not perform well for $k \leq 30$.

In the experiments for Wisconsin data set with rare class, Table B.7 show that no single algorithm dominates, but our new algorithms performed better than LOF , COF , INFLO and DBCOD.

For iris data set with planted outliers (see Table B.9), ODMRW achieves the best performance for all values of k. RADA, ODMR, ODMRS, ODMRD and RBDA all have similar performances. LOF, COF and INFLO do not work well for this data set since they hardly detect any outlier when $m \leq 40$.

For ionosphere data set with planted outliers (see Table B.10), ODMRW performs better than the other algorithms for all values of k. DBCOD, LOF, COF and INFLO do not work well for this data set since they are not able to detect more than 1 (out of 3) outlier when $m \leq 30$.

For Wisconsin data set (see Table B.11), RADA, ODMR, ODMRS, ODMRW, ODMRD and RBDA achieve the best performances for values of k.

Note that higher value of m_t, recall, and RankPower represent the better performance.

RankPower is a more discriminatory metric. Using this metric, the relative behavior of

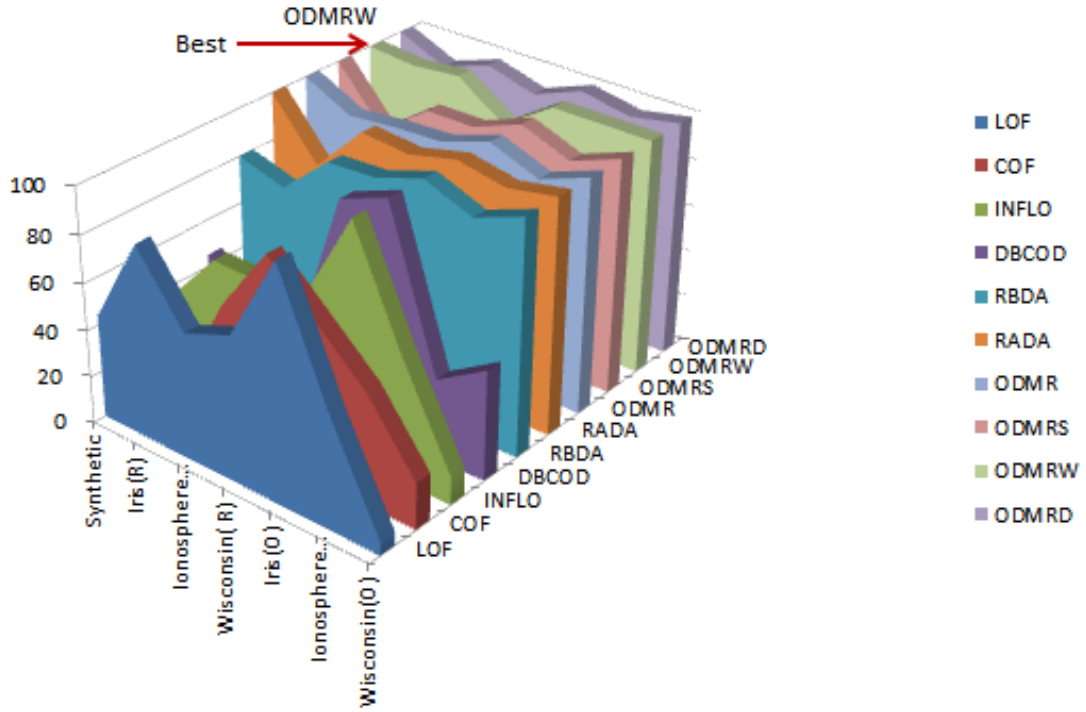

Fig. 3.4: **Overall performance of algorithms for 7 data sets.** - Average performance of LOF, COF, INFLO, DBCOD, RBDA, RADA, ODMR , ODMRS, ODMRW, and ODMRD for all data sets by using different values of k. Y axis presents the ratio of RankPower of an algorithms versus the best RankPower of all the algorithms.

algorithms can be summarized as:

$$\begin{aligned} \text{ODMRW} &\geq \text{ODMRD} \geq \text{ODMR} \geq \text{RADA} \\ &\geq \text{ODMRS} \geq \text{RBDA} \geq \text{LOF} \geq \text{DBCOD} \geq \text{INFLO} \geq \text{COF} \end{aligned}$$

where by "$\geq$" we indicate a better performance.

Table 3.1, which summarizes the performance over all values of k, is obtained by using *normalized* RankPower. RankPowers are normalized to fall in the scale [0, 100], where 0 means that the RankPower of the algorithm is least, and 100 corresponds to the largest RankPower. Values of k were chosen between 5% to 10% of the size of datasets. It can be seen that ODMRW has the best overall performance.

Table 3.1: Summary of LOF,COF, INFLO, DBCOD, RBDA, RADA, ODMR , ODMRS, ODMRW, and ODMRD for all experiments.

Dataset	LOF	COF	INFLO	DBCOD	RBDA	RADA	ODMR	ODMRS	ODMRW	ODMRD
Synthetic	44	27	36	45	80	100	100	100	100	100
Iris(r)	82	17	59	26	74	76	90	77	98	91
Inosphere(r)	54	57	58	26	92	97	92	92	100	98
Wisconsin(r)	62	86	61	92	93	95	93	92	86	92
Iris(o)	100	67	100	100	100	100	100	100	100	100
Inosphere(o)	49	48	53	34	92	96	92	92	100	96
Wisconsin(o)	6	20	12	47	100	100	100	100	100	100
Summary	57	46	54	53	90	95	95	93	98	97

Note: Numbers represent the average performance rank of the algorithms; a larger value implies better performance. Data set with 'r' in the parentheses represents the data set with rare class. And data set with 'o' in the parentheses represents the data set with planted outliers.

3.6 Conclusions

The performance of an outlier detection algorithm based on rank alone is highly influenced by cluster density variations. Furthermore, by definition, ranks use the relative distances

and ignore the ‘true’ distances between the observations. This motivated us to develop new outlier detection algorithms that utilize rank as well as distance information.

Extensive evaluations on synthetic and real datasets have demonstrated that the overall performance of each of the new algorithms (ODMR and other variants) is significantly better than previously known algorithms. Among the new variants of algorithms, ODMRW performed best, perhaps due to the greater weightage placed on distance.

CHAPTER 4
DETECTION OF ANOMALOUS TIME SERIES

In this chapter, we address the problem of detecting an anomalous time series with respect to entire time series data set. To the best of our knowledge existing methods for anomaly detection use a single measure, failing to capture several varieties of anomalies that occur in practical applications. This motivates our attempt to develop an anomaly detection method based on a combination of multiple distance measures. The resulting “MUDIM" algorithm is successful in detecting anomalies in most of the data sets we have examined, with much better success rates than other algorithms.

This chapter is organized as follows: in Section 4.2, we review existing approaches and evaluate them. In Section 4.3, we propose a multiple distance measure based detection method and evaluation procedure. In Section 4.4, we present experimental details and results; Section 4.5 summarizes the results.

4.1 Problem Definition Revisit

Let us revisit the problem: suppose we are given a time series data set $\mathcal{D} = \{x_i(t)|1 \leq t \leq n; i = 1, 2, \ldots, m\}$, where $x_i(t)$ represents the data object of the ith time series at time t, n is the length of the time series, and m is the number of time series in the dataset. The goal is to calculate $O(x_i)$, the outlierness of a series x_i, and $O_{threshold}$ a threshold such that if x_i is an anomalous series, then $O(x_i) \geq O_{threshold}$; otherwise $O(x_i) < O_{threshold}$.

4.2 Existing Approaches - A Revisit

Identification of an anomalous time series is an important data mining task with many applications such as detecting fraud in credit card datasets, abnormal events in ECG data, network intrusion, electronic device measurement, image processing and astronomical analysis. For example, this technique can be used to detect light curves' outliers within astronomical data [53]. For financial data, consider time series in Figure 4.1 representing the stock prices of the oil and gas companies for 2010 to 2012 in which the behavior of one series (dashed red line) is different from the rest. Another example is for solenoid current measurement. In Figure 4.2, the normal time series correspond to the data measured during the normal operations of the valves and the anomalous series is measured during different faulty operations of the valves. The goal of this study is to detect such anomalous behavior in a collection of time series.

Viewed as a vector a time series is generally of very large dimensionality. Consequently, the first step is to obtain a compact representation to capture important information contained in the series. Three main categories of approaches have been identified, by Ding *et al.*[16] and Chandola *et al.* [10], to reduce dimensionality of time series: *model based, data-adaptive*, and *non-data adaptive* approaches. The compact representations obtained using these approaches must then be compared using suitable distance measures, which have been categorized into three groups [16]:

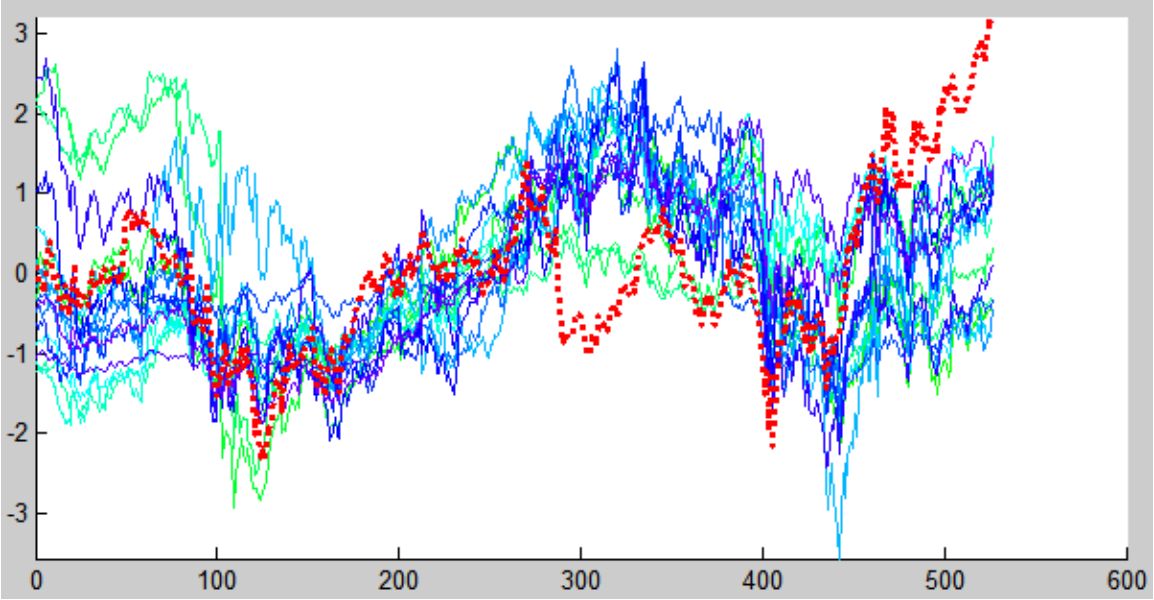

Fig. 4.1: **Stock prices for some oil and gas companies and an outlier series.** - The anomalous series is stock price of Walmart. Red dots represents the anomalous series.

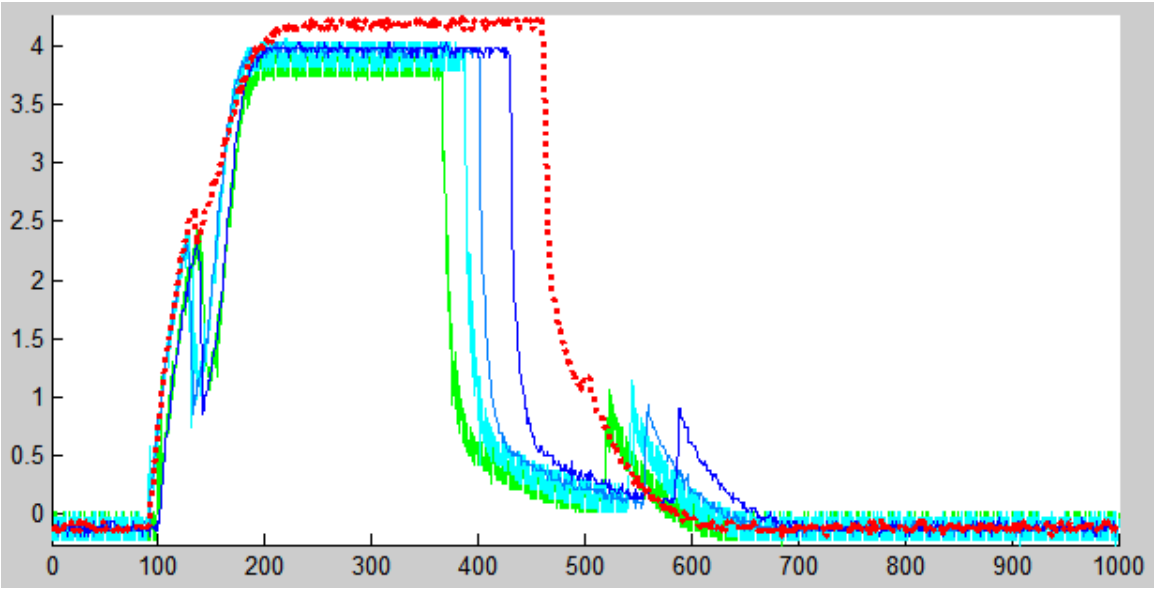

Fig. 4.2: **Solenoid current measurements on Marotta MPV-41 series valves.** - The anomalous series are measured during different faulty operations of the valves. Red dots represents the anomalous series.

- The lock-step measures are based on one-to-one mapping between two time series. Examples include: Cross Euclidean distance (EUC) [21], the Cross Correlation Coefficient-based measure, defined as $\sqrt[2]{2(1 - \text{corrcoef}(d_x, d_y))}$ [4], SameTrend (STREND), and standard deviation of differences (DIFFSTD) (defined in Section 4.3.2)).

- The elastic measures are based on one to many mapping between two time series, e.g., Dynamic time warping (DTW) [3] [38], and Edit Distance on Real sequence (EDR) [14].

- Measures in the transformed space, includes TQuEST [1] distance and Spatial Assembling Distance SpADe [15]. Other examples include Symbolic Aggregate approXimation (SAX) proposed by Keogh and Lin [41] with and without sliding window (SAXSL and SAXNW respectively); SAX with bag-of-pattern (SAXBAG) [46], Discrete Wavelet Transform [8], and Discrete Fourier Transform [21].

Wei *et al.* [62] proposed an algorithm to find unusual shapes by using SAX presentation to speed up the process for selecting the best candidates. Lin *et al.* proposed distance between bag-of-pattern [46] based on frequencies of each unique SAX words to find the similar time series, which can be also applied for anomalous time series detection. Keogh *et al.* [38] suggested indexing techniques for dynamic time warping by using their proposed lower bounding measure, which can be used for detecting distance measures of time series. Protopapas [53] computed outlier measure as the average of correlations between time series. Chan [8] showed that Euclidean distance on Haar transformed domain can be effective for finding time series matches and it outperformed Discrete Fourier Transformation (proposed by Faloutsos [21]). Bu *et al.* [6] applied Haar wavelet transform on time series and built an augmented trie to find top k discords in time series database. In Table 4.1, we summarize the Pros and Cons for some popular distance measures mentioned above. It is clear that no single measure is able to detect all types of anomalous series.

Table 4.1: Distance Measures Pros and Cons

Measures	Pros	Cons
EUC[21] CCF [4] DIFFSTD	Easy to implement; Computationally efficient;	Lock-step measure; Normal series with time lagging cause problems;
Dynamic Time Warping[38]	Elastic measure; Comparison with time lagging	Small deviations may not be detected;
Discrete Fourier Transform[21] (DFT)	Good in detecting anomalies in frequencies domain; Normal series with time lagging do not cause problem;	Small deviations may not be detected; Cannot detect anomalies with time lagging;
Discrete Wavelet Transform[8] (DWT)	Good in detecting anomalies in frequencies domain;	Small deviations may not be detected; Sensitive to time lagging;
SAX with sliding window [41] (SAXSL)	Tolerates noise, as long as its standard deviation is ‘small’;	May not detect abnormal subsequence of shorter length than feature window size; Normal series with time lagging cause problems;
SAX without sliding window [41] (SAXNW)	Tolerates noise, as long as its standard deviation is ‘small’	May not detect abnormal subsequence of shorter length than feature window size; Small deviations may not be detected Normal series with time lagging cause problems
SAX with bag-of-pattern[46] (SAXBAG)	Tolerates noise, as long as its standard deviation is ’small’ Normal series with time lagging do not cause problem.	Cannot detect anomalies with time lagging; Cannot detect anomalous series with similar frequencies but different shapes;

Table 4.2: Correlation coefficient matrix between distances of each pair of series of 30 data sets correspondingly based on different measure.

	SAXSL	SAXNW	SAXBAG	DTW	DIFFSTD	STREND	WAVELET	FOURIER
SAXSL	0.000	0.479	0.281	0.382	0.594	0.576	0.594	0.422
SAXNW	0.479	0.000	0.263	0.499	0.687	0.429	0.685	0.476
SAXBAG	0.281	0.263	0.000	0.374	**0.339**	0.345	0.317	0.389
DTW	0.382	0.499	0.374	0.000	0.718	0.295	0.698	0.653
DIFFSTD	0.594	0.687	0.339	0.718	0.000	0.482	0.938	0.672
STREND	0.576	0.429	0.345	0.295	0.482	0.000	0.468	0.348
WAVELET	0.594	0.685	0.317	0.698	0.938	0.468	0.000	0.653
FOURIER	0.422	0.476	0.389	0.653	0.672	0.348	0.653	0.000
AVERAGE	0.476	0.503	0.330	0.517	**0.633**	0.420	0.622	0.516

FEATURE1	FEATURE2	FEATURE3	PARTIAL CORRELATION
DIFFSTD	SAXBAG	SAXSL	0.499
DIFFSTD	SAXBAG	SAXNW	0.126
DIFFSTD	SAXBAG	DTW	0.240
DIFFSTD	SAXBAG	STREND	**0.096**
DIFFSTD	SAXBAG	WAVELET	0.265
DIFFSTD	SAXBAG	FOURIER	0.196

Feature1	Feature2	Correlation Coefficient	Feature1	Feature2	Correlation Coefficient	MAX
DIFFSTD	SAXSL	0.594	SAXBAG	SAXSL	0.281	0.594
DIFFSTD	SAXNW	0.687	SAXBAG	SAXNW	0.263	0.687
DIFFSTD	DTW	0.718	SAXBAG	DTW	0.374	0.718
DIFFSTD	STREND	0.482	SAXBAG	STREND	0.345	**0.482**
DIFFSTD	WAVELET	0.938	SAXBAG	WAVELET	0.317	0.938
DIFFSTD	FOURIER	0.672	SAXBAG	FOURIER	0.389	0.672

4.3 Outlier Detection Based on Multiple Distance Measures - MUDIM

4.3.1 Measure Selection

As described earlier, no single measure is capable of capturing different types of perturbations that may make a series anomalous. Consequently it makes sense to include more than one measure. An obvious choice is to select measures that are orthogonal to each other. We have tried several methods to select the best combination of measures. First we describe the method based on correlation coefficient. The heuristic is to select measures that are least correlated with each other. The correlation coefficients of distances based on different measures are calculated and selection is made using the heuristic. We used the following steps:

1. Selected 30 data sets as testing set, which is denoted as $D_T = \{D_m | 1 \leq m \leq 30\}$ (see starred data sets in Tables 4.3 and 4.4) from multiple application domains such as finance, electronics, astronomic, and image recognition, including some synthetic data sets. The details of each data set is provided in Table 4.3.

2. We selected 8 distance measures: SAXSL, SAXNW, SAXBAG, DTW, WAVELET, FOURIER, DIFFSTD AND STREND. The last two measures are introduced in the following paragraphs. This collection is denoted as $\mathcal{F}$.

3. Then we apply each distance measure for every data set, and calculate the distance between every possible pair of series in the data set. Let $Dist_{m,n}(i,j)$ be the distance between ith and jth series in the data set D_m by using nth distance measure in $\mathcal{F}$, where $1 \leq m \leq 30, 1 \leq n \leq 8$. The correlation coefficient between pth and qth distance measures for the data set D_m is denoted as $CCF_m(p,q) = corrcoef(Dist_{m,p}(i,j), Dist_{m,q}(i,j))$, where $i, j \leq |D_m|$.

4. Let the average correlation coefficient between pth and qth measure $\in \mathcal{F}$ for all data sets be denoted as $CCF(p,q) = \frac{\sum_{m=1}^{30} CCF_m(p,q)}{30}$.

5. The correlation coefficient matrix between different measures is obtained, as shown in Table 4.2, and average correlation coefficient $CCF(p)$ is calculated as $\frac{\sum_{q \in \mathcal{F}, q \neq p}^{|\mathcal{F}-1|} CCF(p,q)}{|\mathcal{F}|-1}$.

6. Select the measure which has the highest CCF(p) values. This measure is DIFFSTD.

7. Select the measure with the lowest correlation coefficient with DIFFSTD. It is SAXBAG.

8. In order to get the real relationship between DIFFSTD and SAXBAG, a partial correlation can be calculated between DIFFSTD and SAXBAG, controlling with third measure; as Table 4.2 shows with STREND, partial correlation coefficient between DIFFSTD and SAXBAG is the minimum (0.096).

9. Calculate CCF values for a measure versus DIFFSTD and SAXBAG respectively. Select the larger value. Repeat this for all other measures. Finally select the smallest of them. The corresponding measure is our next selected measure.

10. Finally, DIFFSTD, SAXBAG, and STREND are our potential candidates.

Our second method to find a subset of above described measures is the exhaustive method. We tried our algorithms (introduced in Section 4.3.2) based on each possible combination consisting of two to four measures on 30 data sets introduced above, and then selected the combination which achieves the best performance. The results show that: combination of two or four measures is generally worse than a combination of three measures. The combination consisting of DIFFSTD, SAXBAG, and STREND achieves the best performance in our experiments.

4.3.2 Anomaly Detection Based on Selected Measures

The correlation based approach selected three measures: (a) SAXBAG, (b) Same Trend (STREND), and (c) Standard deviation of differences between two time series (DIFFSTD). These three measures are described below, with a brief discussion that further justifies their selection.

- SAXBAG (proposed by Lin *et al.* [46]): Given a time series, a subsequence of size ℓ is obtained using a sliding window. Each subsequence is reduced to w-dimensional discrete representation in two steps. First, the subsequence is further divided into $u = \frac{\ell}{w}$ equal size sub-subsequences and the average value of the data falling within a sub-subsequence is evaluated. In the second step, each average is discretized to a symbol, from a predetermined set of symbols, using equal probability intervals approach. Figure 4.3(a) illustrates these concepts useing three symbols: a, b, and c.

- STREND: This is a new measure that we propose. For each i, we calculate the difference $x'_i(t) = x_i(t+1) - x_i(t), t \in [1..n-1]$ and define

$$S_{i,j}(t) = \begin{cases} 1 & \text{if } x'_i(t) \cdot x'_j(t) > 0 \\ -1 & \text{if } x'_i(t) \cdot x'_j(t) < 0 \\ 0 & \text{otherwise.} \end{cases}$$

  Clearly, $S_{i,j}(t)$ indicates whether or not $x_i(t)$ and $x_j(t)$ change in the same direction at time t. The aggregate measure, over the entire length, n, of the time series is evaluated as

$$dist(i,j) = 1 - \sum_{t \in [1..n-1]} S_{i,j}(t)/(n-1). \tag{4.1}$$

- DIFFSTD is the standard deviation of differences between two time series, i.e., if $\delta_{i,j}(t) = \|x_i(t) - x_j(t)\|$, and $\mu_{i,j} = \sum_t \delta_{i,j}(t)/n$, then the new distance is defined

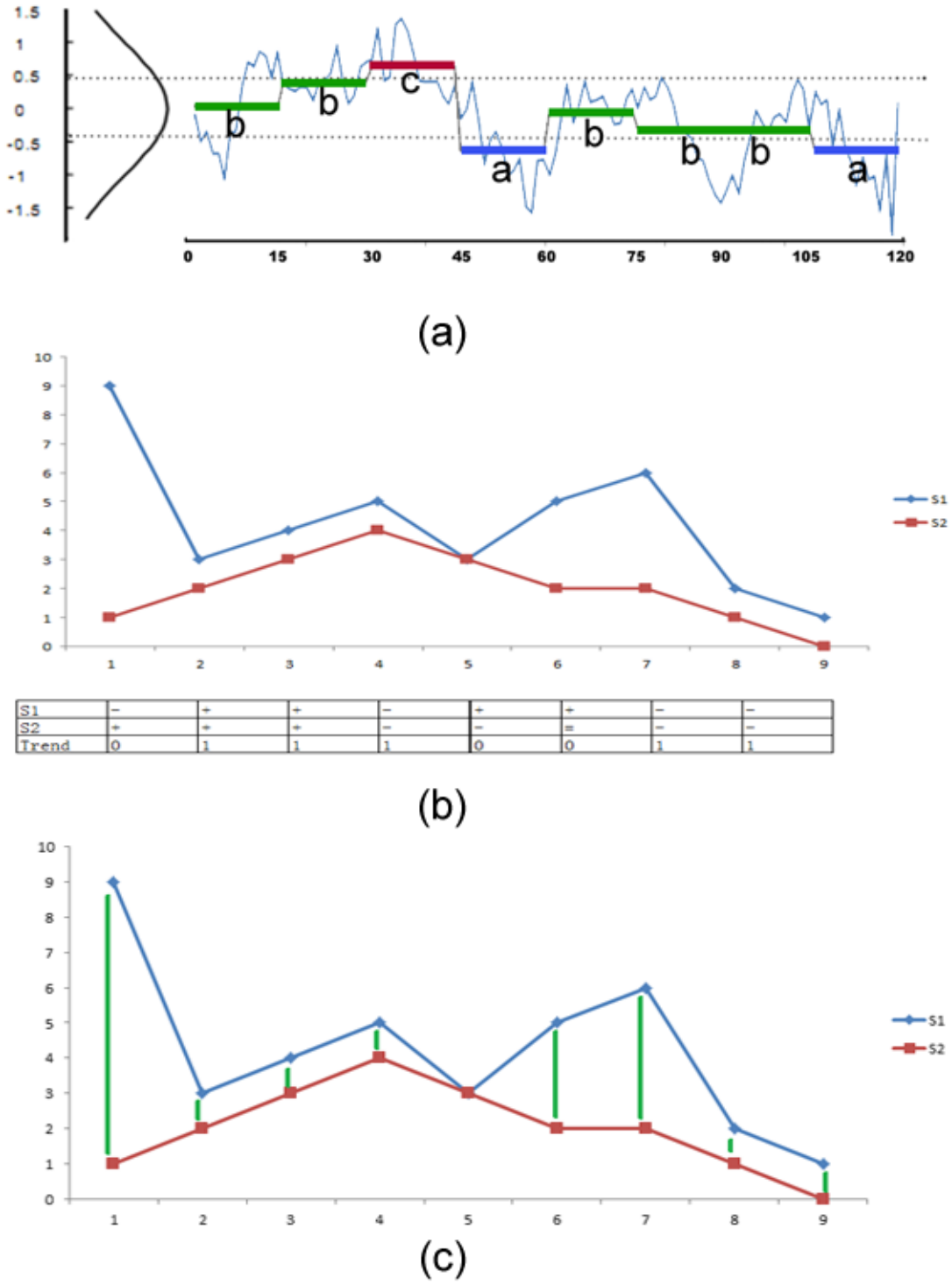

Fig. 4.3: **Illustrations for three key measures of MUDIM** - (a) Illustrates how a SAX word, in a sliding window, is generated. SAXBAG counts the frequencies of such words in the word-sequence; (b) illustrates STREND (table under the figure shows $S_{i,j}(t)$'s); and (c) illustrates DIFFSTD (vertical lines show the differences between two time series x_1 and x_2).

as dist$(x_i, x_j) = \sqrt{(\sum_t (\delta_{i,j}(t) - \mu_{i,j}(t))^2/n}$. This measure is widely used in pair trading in the financial field.

These measures capture different aspects of the time series, and a combination of these gives a comprehensive measure of how isolated is a time series from others in the comparison set. SAXBAG captures behavior of a time series using a histogram of possible patterns, STREND identifies the degree of synchronization of a series compared with another series, and DIFFSTD measures the amount of deviation, as illustrated in Figure 4.3. Combining these three metrics produces a comprehensive and balanced distance measure that is more sensitive than individual measures.

From another perspective, SAXBAG addresses the “signature of a single time series" whereas DIFFSTD and STREND are inter-time-series-measures; STREND focuses on the synchronization of two time series whereas DIFFSTD focuses on the magnitude.

4.3.3 Normalization

After all three distance measures are calculated, they need to be normalized and combined together. The ranges of selected measures are very different; for example, the range of STREND is from 0 to 1, and the range of SAXBAG is from 0 to $\sqrt{2 \times l^2}$ where $l =$ the number of words obtained from the entire series. So the three distance measures have to be normalized in order to be processed further without any bias due to the different ranges.

Different normalization methods were tested including (1) linearly mapping in [0,1] range; (2) removing 5% of the extreme observations on either end, then linearly mapping to [0,1] range; (3) Dividing observations by trimmed mean (excluding 5% on either end). Among all these normalization methods, the last normalization method appears to perform best[1] . The normalized distances between the ith and the jth series, based on SAXBAG, STREND, and DIFFSTD are denoted as $\text{dist}'_s(i,j)$, $\text{dist}'_t(i,j)$, and $\text{dist}'_f(i,j)$, respectively.

[1] An extreme observation can strongly affect the mean and the standard deviation of a data set. To achieve ‘improved’ performance and robustness, use of trimmed mean is appropriate.

4.3.4 Assignment of Weights to Selected Measure

Anomalousness of a time series can be measured in aforementioned three ways. However, it is quite possible that all three are not equally effective in detecting anomalousness. Consequently it makes sense to assign the highest weight to that measure which is more effective. We calculate three weights adaptively, using the rank based approach discussed in the previous Chapter. Different weights are assigned for different measures of each series in a data set. The weight associated with a measure ℓ of the ith series in a given data set, weight$'_\ell(i)$, is computed as follows:

1. For the ith series, find its k-nearest-neighbors (kNN) using the distance measure.
2. Based on kNN, find RBDA value, $\mathcal{O}_k(p)$, of ith series using equation 2.1.
3. weight$'_\ell(i) = \mathcal{O}_k(p)$.

The advantage of our method for assignment of weight is its adaptability since rank based algorithm automatically adjust the weights according to its anomalousness. Details can be found in Algorithm MUDIM.

4.3.5 Decision for Anomalous Series

Getting the anomaly score for each series is not the final step, because there is one last question left: which series is anomalous? This question we have again, can be answered by our previous work in Chapter 3. We have two solutions: threshold based and clustering based.

The basic idea of the first method is to assign the anomaly score for each series, then decide the threshold for anomaly determination. The anomaly score for each series is calculated based on distances and weights of the three measures mentioned in previous

section.

$$A_i = \sqrt{\sum_{\ell \in \{s,t,f\}} (\text{dist}'_\ell(i))^2 \times \text{weight}'_\ell(i)}. \tag{4.2}$$

Then we sort A_i in descending order and calculate the average and standard deviation of the lower 80% of sorted A_i observations. If the ith series has an A_i larger than three standard deviation over average, it is declared as an anomalous series.

The second method is to make decisions via clustering. This idea can be applied in at least two ways: 1) Apply the clustering algorithm $\mathcal{NC}(\ell, m^*)$ described in Chapter 3 in 3-dimensional space (using all three features of a series), if a series belongs to a cluster, then it is declared as normal, otherwise potentially anomalous; 2) Apply it in each feature space separately. If a series belongs to one cluster in all three feature spaces, then it is declared as a normal series, otherwise it is potentially anomalous. Finally, each potentially anomalous series can be compared with normal series. For instance, if anomalousness is 3 standard deviation above than that of a normal series, then it can be considered anomalous.

After empirical comparison of above two methods, the first one appear to be the better, since the latter method results in more false positives.

All the steps of the resulting MUDIM algorithm are shown in Algorithm MUDIM.

4.3.6 Evaluation Methods

All time series in the given data set are normalized to have zero mean and standard deviation equal to one, as suggested by Keogh [40], who writes “both series must be normalized to make the meaningful comparisons between time series".

In the experimental study described below, we use the widely used *RankPower* comparison metric [57].

RankPower: Suppose we know the anomalous series in the dataset. The set of all such series is denoted by $O \subset \mathcal{D}$. A good algorithm should declare a series anomalous if it

Algorithm 1 MUDIM algorithm.

Require: a positive integer k and a time series dataset $\mathcal{D}$.

Ensure: outliers in $\mathcal{D}$.

Step 1. Calculate the distances between ith and jth time series using SAXBAG, STREND, and DIFFSTD, denoted as $\text{dist}_s(i,j)$, $\text{dist}_t(i,j)$, and $\text{dist}_f(i,j)$, respectively.

Step 2. For $\ell = s, t, f$ normalize the raw distance to be between 0 and 1 as follows:

$$\text{dist}'_\ell(i,j) = \frac{\text{dist}_\ell(i,j)}{mean([5\% \ldots 95\%] \text{ of sorted list of } \text{dist}_\ell(i,j))}$$

Step 3. The weight for ith series for normalized ℓ feature for $\ell \in \{s,t,f\}$, is as $\text{weight}'(i)$, can be calculated as:

$$\text{weight}'_\ell(i) = W_k(i); \text{ where } W_k(i) \text{ is calculated by equation } 3.1 \text{ in Chapter } 3.$$

$$W_k(i) = \mathcal{O}_k(i) \times \frac{\sum_{q \in \mathcal{N}_k(i)} d(q,i)}{|\mathcal{N}_k(i)|}$$

Step 4. For all $i \in \mathcal{D}$, find k-nearest neighbors of the i^{th} time series using $\text{dist}'_\ell(i,j)$ for $j = 1, \ldots, m;\ j \neq i; \ell \in \{s,t,f\}$.

$$\text{dist}'_\ell(i) = \frac{\sum_{j \in \mathcal{N}_k(i)} \text{dist}'_\ell(i,j)}{|\mathcal{N}_k(i)|}$$

where $\mathcal{N}_k(i)$ denotes the set of k-nearest neighbors.

Step 5. Calculate the anomaly score, A_i, for the i^{th} time series as combined distance based on weighted distances:

$$A_i = \sqrt{\sum_{\ell \in \{s,t,f\}} (\text{dist}'_\ell(i))^2 \times \text{weight}'_\ell(i)}.$$

Step 6. Sort A_i in descending order and calculate the average and standard deviation of lower 80% of sorted A_i observations. Any ith series with an A_i larger than three standard deviation over the average is declared to be an anomalous series.

belongs to O, i.e., we expect that the algorithm will assign large anomaly scores to series in O and smaller values to series $\in \mathcal{D} - \mathcal{O}$. To quantify this expectation we define RankPower as follows. First we sort series in descending order of magnitude of their anomaly scores. The series with largest score is assigned the rank 1, and the series with the smallest value is assigned rank = m. We now define:

$$\text{RankPower } = \frac{|O|(|O|+1)}{2\sum_{i \in O} R_i}.$$

Clearly, if all series in O are ranked between 1 and $|O|$, the RankPower takes its largest possible value, 1. Moreover, if RankPower of Algorithm I is larger than the RankPower of Algorithm II, then Algorithm I is considered to be better in detecting anomalous series.

4.4 Datasets and Results

We have used 47 data sets, consisting of our three synthetic datasets and 44 modified real datasets from all kinds of application domains, to compare performances of algorithms, including MUDIM. In our experiments, three parameters of SAXBAG, the size of subsequence, the length of SAX word and the number of symbols, were 20, 5, and 5, respectively.

4.4.1 Datasets

Synthetic datasets are designed to introduce typical time series problems such as time lagging and amplitude differences (see Figure 4.4). The real datasets come from different application domains, such as synthetic, finance, electronic, image recognition and video surveillance. We augment these real datasets by introducing one or more anomalous series.

Details about the normal series and anomalous series in each data set are given in Table 4.3 and 4.4. Since some data sets were originally designed for classification problems, we modified them for our anomaly detection experiments by selecting all time series from one

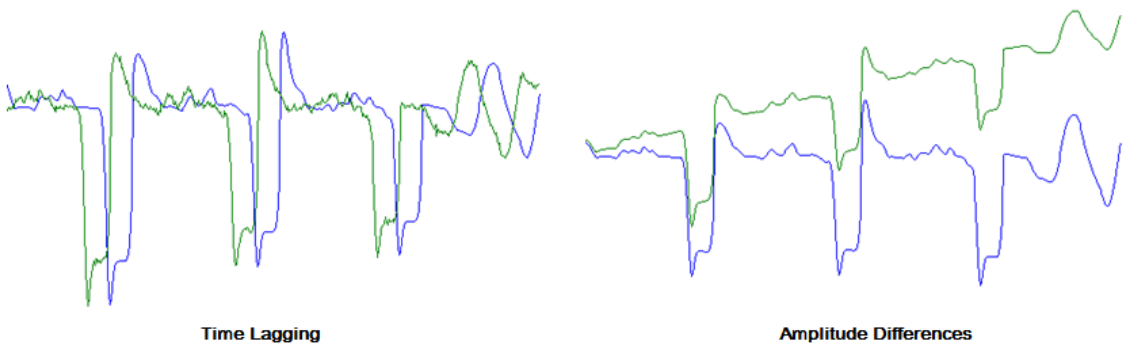

Fig. 4.4: **Typical time series problems** - Time lagging: two series $T_a(t)$ and $T_b(t)$, $T_a(t) = T_b(t + x)$; Amplitude differences: $T_a(t) <> T_b(t)$

class and choosing a few time series from another class that are very different from the most of the series of the first class, thus making them anomalous.

Table 4.3: Summary of all time series sets. Similarity of normal series represents that how similar normal series look like to each other.

Dataset	Domain	Length	#OfSeries	#OfOutliers	Normal Series			Anomalous Series					
					Lagging	Amplitude	Similarity	Lagging	Amplitude	Dissimilarity to normal series			
										<10%	<30%	<60%	>=60%
*SYN1	Synthetic	501	14	2	Yes		High						Yes
SYN2	Synthetic	128	30	2	Yes	Yes	High		Yes			Yes	
*SYN3	Synthetic	500	7	1		Yes	High						Yes
*STOCKS	Finance	527	18	1		Yes	Medium		Yes			Yes	
*OPMZ	Finance	629	6	1		Yes	Low		Yes				Yes
*EMP3	Astronomic	201	11	3	Yes		High		Yes			Yes	
*MOTOR	Electronic	1500	21	1	Yes		High		Yes		Yes		
*POWER	Power	672	51	7		Yes	Medium		Yes		Yes		
*TEK140	Electronic	1000	5	1	Yes		Medium	Yes	Yes	Yes			
*TEK160	Electronic	1000	5	1	Yes		Medium	Yes	Yes	Yes			
*TEK170	Electronic	1000	5	1	Yes		Medium	Yes	Yes	Yes			
*SHAPE1	Image	1614	21	1	Yes		Medium	Yes	Yes		Yes		
*SHAPE2	Image	1614	21	1	Yes		Medium	Yes	Yes		Yes		
*ADIAC1	Image	2801	7	1		Yes	Medium		Yes				Yes
ADIAC2	Image	176	43	1			Exact		Yes	Yes			
*BEEF1	Unknown	470	7	1		Yes	Medium		Yes	Yes			
*COFFEE1	Unknown	286	15	1			High		Yes	Yes			
*COFFEE2	Unknown	286	15	1			High		Yes	Yes			
*CBF1	Unknown	128	11	1	Yes	Yes	Low		Yes				Yes
*CBF2	Unknown	128	13	1	Yes	Yes	Low		Yes				Yes
*WORDS1	Text Recognition	2153	14	1	Yes	Yes	Medium		Yes			Yes	
*WORDS2	Text Recognition	270	14	1	Yes	Yes	Medium	Yes	Yes			Yes	
*WAFER1	Semiconductor	152	21	2	Yes	Yes	High	Yes	Yes				Yes
WAFER2	Semiconductor	152	84	4	Yes	Yes	High	Yes	Yes				Yes
ECG1	Medical	96	52	3			High	Yes	Yes				Yes
ECG2	Medical	96	55	2			High	Yes	Yes				Yes
FACEALL1	Face Recognition	131	74	2	Yes	Yes	Medium	Yes	Yes				Yes
FACEALL2	Face Recognition	131	140	2	Yes	Yes	Low	Yes	Yes				Yes

Table 4.4: Summary of all time series sets (Cont).

Dataset	Domain	Length	#OfSeries	#OfOutliers	Normal Series			Anomalous Series					
					Lagging	Amplitude	Similarity	Lagging	Amplitude	Dissimilarity to normal series			
										<10%	<30%	<60%	>=60%
*SBANK	Finance	426	12	1		Yes	Low		Yes				Yes
*SMINE	Finance	426	10	1			High		Yes				Yes
*YOGA1	Image	426	11	1	Yes		High		Yes			Yes	
YOGA2	Image	426	22	1	Yes		High		Yes		Yes		
*SWEDL1	Image	509	14	1	Yes		Medium		Yes				Yes
SWEDL2	Image	128	36	1	Yes		Medium		Yes		Yes		
FISH1	Image	463	22	1	Yes		High		Yes	Yes			
FISH2	Image	463	26	1	Yes		High		Yes		Yes		
*OLIVE1	Unknown	570	9	1			High		Yes	Yes			
*OLIVE2	Unknown	570	14	1			High		Yes	Yes			
TWOPT1	Synthetic	128	104	3	Yes	Yes	High	Yes	Yes	Yes			
TWOPT2	Synthetic	128	128	3		Yes	Medium		Yes				Yes
*LIGHTING1	Synthetic	637	9	1	Yes	Yes	Low	Yes	Yes			Yes	
*LIGHTING2	Synthetic	637	12	1	Yes	Yes	Low	Yes	Yes			Yes	
*FACE41	Face Recognition	350	14	1	Yes		High	Yes	Yes				Yes
*FACE42	Face Recognition	350	14	1	Yes		High	Yes	Yes				Yes
GUNPT1	Video surveillance	150	76	1	Yes	Yes	High		Yes	Yes			
GUNPT2	Video surveillance	150	76	1	Yes	Yes	High		Yes	Yes			
GUNPT3	Video surveillance	150	76	1	Yes	Yes	High		Yes	Yes			

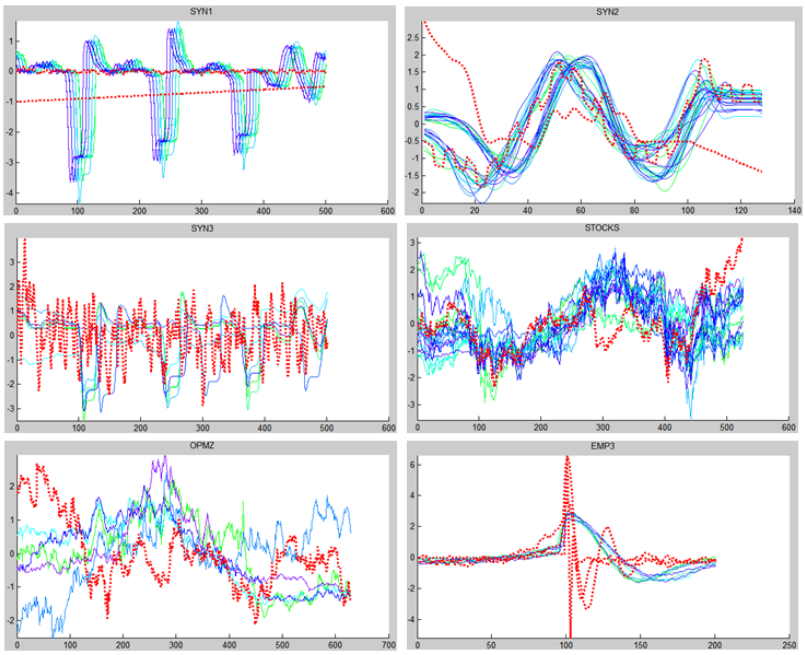

Fig. 4.5: **Plots of time series data set, part 1** - Red dots line represents anomalous series

Some background information about the data sets is given below:

- **Synthetic Dataset 1(SYN1)**. Synthetic dataset 1 contains 14 univariate time series including two anomalous time series. The length of each time series is 500. The two anomalous time series have shapes considerably different from the others.

- **Synthetic Dataset 2(SYN2)**. Synthetic dataset 2 contains 30 time series including two anomalous time series, each of length 128. The normal time series consist of two dissimilar groups, but the two anomalous series do not belong to either group.

- **Synthetic Dataset 3(SYN3)**. Synthetic dataset 3 contains 7 time series including one anomalous time series with the length of 500. The dataset contains time series with many types of scaling such as increasing scaling and varying scaling. The anomalous time series is a single (flat) line perturbed by random noise.

- **Stocks(STOCKS)**. This dataset consists of closing prices of 17 oil & gas operation industry stocks and one consumer retailer stock(WMT:Wal-Mart) from January-4th-2010 to February-10th-2012 [66]. All stock prices are aligned by dates and contain

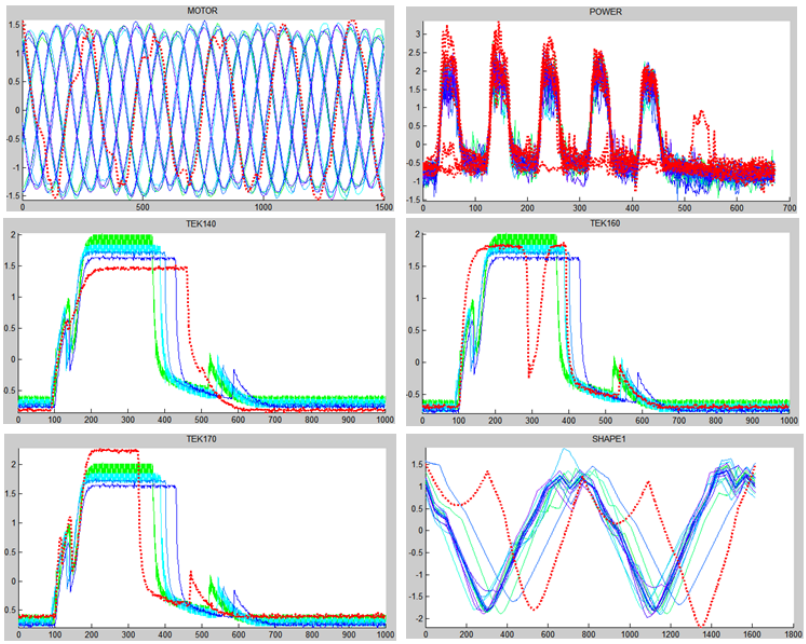

Fig. 4.6: **Plots of time series data set, part 2** - Red dots line represents anomalous series

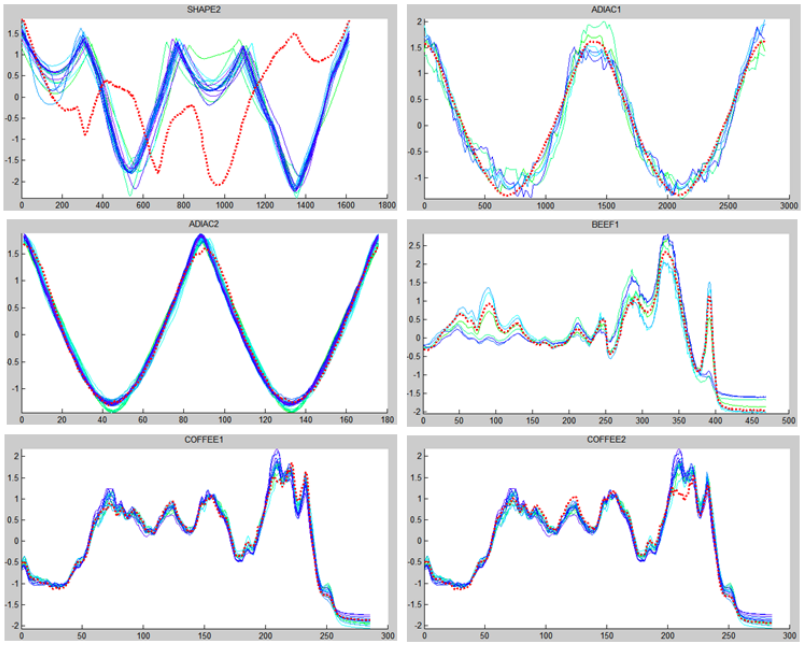

Fig. 4.7: **Plots of time series data set, part 3** - Red dots line represents anomalous series

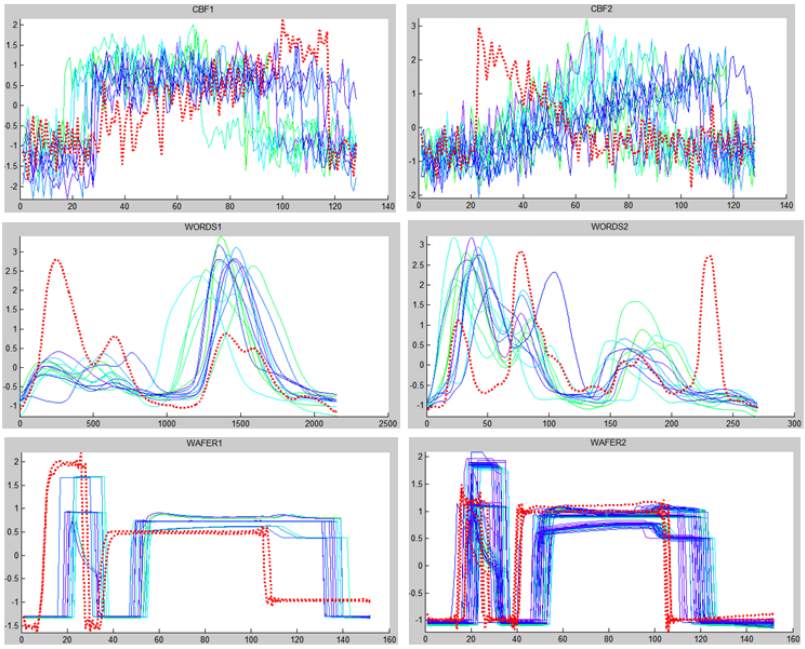

Fig. 4.8: **Plots of time series data set, part 4** - Red dots line represents anomalous series

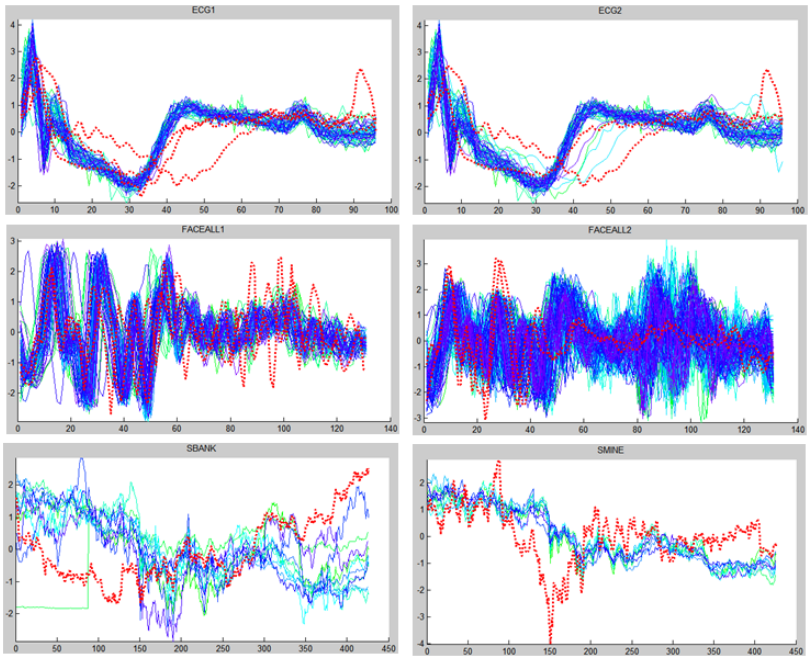

Fig. 4.9: **Plots of time series data set, part 5** - Red dots line represents anomalous series

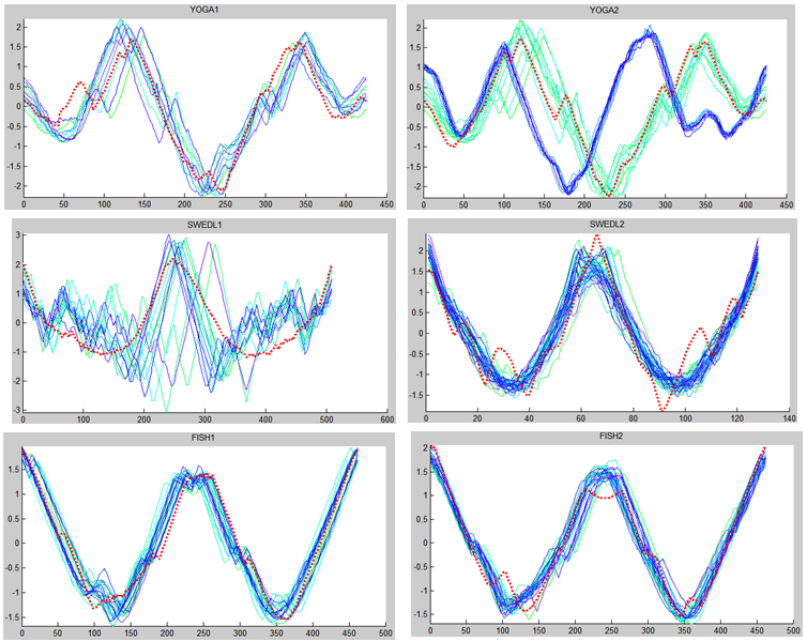

Fig. 4.10: **Plots of time series data set, part 6** - Red dots line represents anomalous series

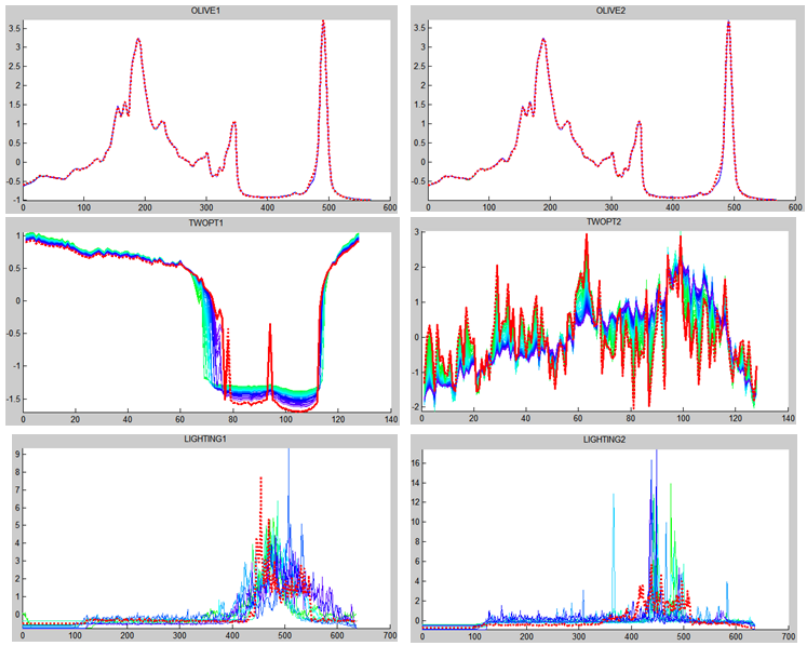

Fig. 4.11: **Plots of time series data set, part 7** - Red dots line represents anomalous series

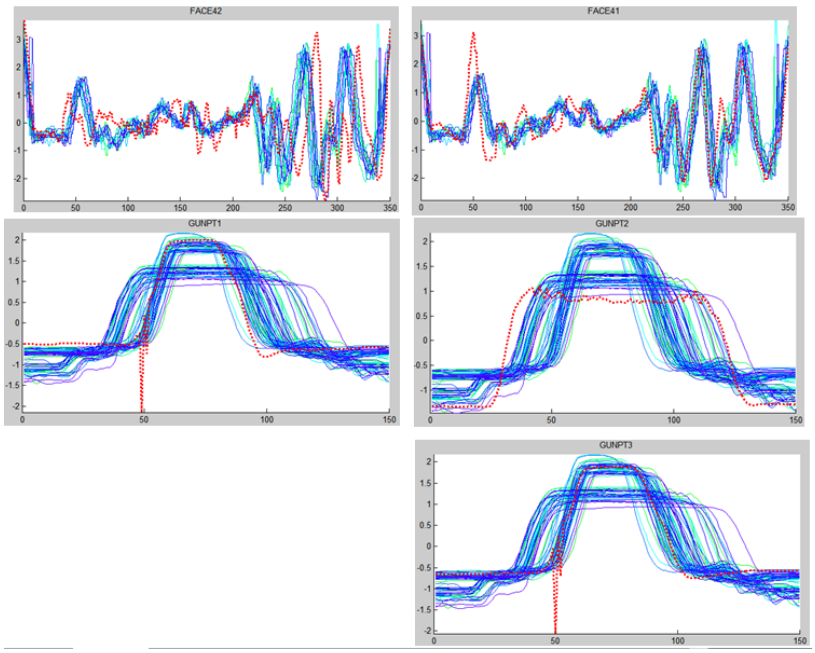

Fig. 4.12: **Plots of time series data set, part 8** - Red dots line represents anomalous series

527 values. The symbols for the 17 stocks in oil and gas industry are APA, APC, BP, CNQ, COP, CVE, CVX, DVN, EOG, HES, IMO, MRO, OXY, STO ,TOT, WNR, XOM. All stock prices are normalized with a mean of zero and standard deviation of one.

- **Commodity Prices(OPMZ)**. This data set contains five commodity prices from [26] and one consumer retailer stock(WMT:Wal-Mart) from October-13th-2009 to April-13th-2012 [66]. Each series contains 629 values. The five commodities are wheat, corn, cocoa, coffee and cotton. All prices are normalized with a mean of zero and standard deviation of one. Since Wal-Mart is not a commodity, the outlier detection method is expected to find Wal-Mart as an outlier in this experiment.

- **Synthetic Lightning EMP (EMP3)**. This data set, from [35], contains 11 time series; 8 of them are from one class and three anomalous series are from another class. Each series contains 201 observations.

- **Motor Current data set(MOTOR)**. Original data set is from [39], and contains 420

time series. 21 were chosen including 1 anomalous time series, and each one consists of 1500 real values. Normal time series are the current signals measured from normal operations of a induction motor. The anomalous time series is obtained from a faulty motor.

- **Power Usage Data(POWER)**. This data set was obtained from UCR [39], and contains 51 time series corresponding to the weekly power consumption measured every 15 minutes at a research facility from week 2 to week 52 in 1977. Each time series contains 672 values, and the anomalous time series represent the power consumption during the weeks with a holiday or special event.

- **NASA Valve Data(TEK140, TEK160 and TEK170)**. This data set was also obtained from UCR [39]. The data values are solenoid current measurements on a Marotta MPV-41 series valve which is on and off under various test conditions in a laboratory. The normal time series correspond to the data measured during the normal operations of the valves; the time series data measured during a faulty operation of the valve is considered an anomaly. Three data sets T14, T16, and T17 all contain 5 time series of which 4 are normal and one is anomalous. The anomalous time series were measured during different faulty operations of the valves.

- **Shape(SHAPE1 and SHAPE2)** These data sets were also obtained from UCR [39]. Both consist of 21 time series corresponding to the shapes. The normal time series have the shapes of bones while the anomalous time series has the shape of a cup.

- **Automatic Diatom Identification using Contour Analysis(ADIAC1 and ADIAC2)**. These data sets ,also obtained from UCR [39],describe the contours of different type of diatoms.

- **Words(WORDS1 and WORDS2)**. These data were also obtained from UCR [39] and describe the contours of word images.

4.4.2 Experiment Results

RankPowers of all algorithms on all data sets are summarized in Table 4.5. From this table, we conclude that MUDIM has the best overall performance and its RankPowers are 1 in all cases; i.e; it finds the anomalous series in all data sets. This confirms our intuition that a combination of measures performs better in most cases. We observe that simple methods such as DIFFSTD, DTW, SAXSL, and FOURIER also work well in some, but not all cases. Domain specific analysis is summarized below:

- MUDIM and STREND are good at detecting outliers in financial data sets such as stock prices and commodity prices, perhaps because financial time series can be easily aligned, and variation in amplitudes is a common problem in financial time series (for example, one stock may increase by 1% and another stock may increase by 5% in the same day). Measures such as SAXSL, SAXBAG, DIFFSTD and FOURIER rarely find real outliers in such data.

- Traditional methods, such as DTW, WAVELET and FOURIER, show good performance in many data sets.

Summarizing the results, MUDIM achieves the best results and shows stable detection accuracy in the experiments.

4.5 Conclusion

In this chapter we have tested the performance of popularly used measures for multiple application domains. We observe that a combination of three measures performed the best, i.e., was able to identify anomalous times series in all domains considered in our experiments. The selection of three measures is, based on some preliminary analyses, and also because we wanted to include measures that do not overlap in their detection capabilities.

Table 4.5: RankPowers of algorithms for 47 data sets.

Dataset	SAXSL	SAXNW	SAXBAG	DTW	DIFFSTD	STREND	WAVELET	FOURIER	MUDIM
SYN1	1	1	1	1	1	1	1	1	1
SYN2	0.3	0.5	1	1	1	1	0.429	1	1
SYN3	0.5	0.333	1	0.5	0.5	0.5	0.5	0.5	1
STOCKS	1	0.5	0.067	0.333	0.5	1	1	0.5	1
OPMZ	1	0.5	0.333	0.5	0.5	1	0.5	0.5	1
EMP3	1	1	1	1	1	1	1	1	1
MOTOR	1	0.048	1	0.077	0.059	0.2	0.059	1	1
POWER	0.933	1	0.189	1	1	0.56	1	1	1
TEK140	1	1	1	1	1	0.5	1	1	1
TEK160	1	0.5	0.333	1	1	0.5	1	1	1
TEK170	1	0.5	0.5	1	1	0.5	1	1	1
SHAPE1	1	1	0.125	1	1	1	1	1	1
SHAPE2	1	1	1	1	1	1	1	1	1
ADIAC1	0.143	1	1	0.333	0.167	0.143	0.167	0.5	1
ADIAC2	0.5	1	1	1	0.5	1	0.333	1	1
BEEF1	0.167	0.167	1	0.143	0.143	1	0.143	0.143	0.5
COFFEE1	1	1	1	0.25	0.333	0.111	0.2	0.5	1
COFFEE2	0.03	0.056	0.6	1	0.068	0.016	0.07	0.231	1
CBF1	0.5	1	0.5	0.333	1	1	1	0.333	1
CBF2	1	0.5	1	1	1	0.5	1	0.333	1
WORDS1	0.25	0.333	0.2	1	1	0.2	1	0.5	1
WORDS2	0.333	1	1	1	1	1	1	1	1
WAFER1	0.6	1	1	1	1	1	1	1	1
WAFER2	0.769	0.625	1	0.667	0.625	0.588	0.588	0.303	1
ECG1	1	1	0.273	0.286	1	0.4	1	0.4	1
ECG2	0.6	0.6	0.25	0.214	0.75	0.2	0.75	0.143	1
FACEALL1	0.6	0.375	0.75	0.75	0.6	0.375	0.6	0.75	0.6
FACEALL2	0.75	0.6	1	0.6	0.429	0.3	0.429	0.107	0.75
SBANK	1	0.5	0.167	0.5	0.5	1	1	0.5	1
SMINE	1	1	1	1	1	1	1	1	1
YOGA1	1	1	0.333	0.2	0.5	1	0.5	0.25	1
YOGA2	0.143	0.2	0.333	0.2	0.25	0.2	0.25	0.5	0.5
SWEDL1	0.5	0.2	1	1	0.2	0.25	0.2	1	1
SWEDL2	1	1	1	1	0.5	0.5	0.5	1	1
FISH1	0.5	1	1	0.143	0.333	0.5	0.333	0.25	1
FISH2	1	1	1	1	1	1	1	0.5	1
OLIVE1	0.333	1	0.5	1	1	0.5	1	1	1
OLIVE2	0.2	1	0.5	1	1	0.125	1	1	1
TWOPT1	0.545	1	1	1	0.6	0.545	1	1	1
TWOPT2	1	1	0.286	1	1	0.09	0.24	0.333	0.667
LIGHTING1	0.25	0.25	1	0.333	0.5	0.333	0.5	1	1
LIGHTING2	1	1	0.2	0.333	0.333	0.5	0.5	0.333	1
FACE41	1	1	0.167	1	1	0.091	1	1	0.2
FACE42	1	1	1	1	1	1	1	1	1
GUNPT1	1	0.143	0.25	1	0.2	0.333	0.2	1	1
GUNPT2	1	1	1	0.25	1	1	1	1	1
GUNPT3	1	0.143	0.04	1	0.167	0.05	0.2	1	0.1
Average	0.733	0.714	0.679	0.722	0.686	0.587	0.685	0.711	0.922

CHAPTER 5
ONLINE ANOMALOUS TIME SERIES DETECTION

In this chapter, we extend to anomalous series detection algorithm (MUDIM) to an online version. This approach, akin to control charts, makes it easy to determine when a series begins to differ from other series. Empirical evidence shows that this novel online anomalous time series detection algorithm performs very well, while being efficient in terms of time complexity, when compared to approaches discussed in the literature (Section 5.2). This chapter is organized as follows: two online algorithms are introduced in Section 5.3; experiments are discussed in Section 5.4; conclusions are given in Section 5.5.

5.1 Problem Statement

Suppose a time series data set $\mathcal{D} = \{x_i(t)|1 \leq t \leq unknown; i = 1, 2, \ldots, m\}$, where $x_i(t)$ represents the data object of the ith time series at time t, length of the time series is unknown, and m - the number of time series in the dataset is given. The goal is to calculate $O(x_i(t))$, the outlierness of a series x_i at time t.

5.2 Literature Review

Online detection requires that we detect the anomalous series as soon as any new observation has arrived. For example, the behavior over time of the stock price of a company may exhibit significant deviations from those of other companies in a comparison group, suggesting an underlying problem that should trigger actions by stock traders. In order to carry out such actions, it is important to detect such deviations as they occur in time, else the trader may be too late and may suffer losses due to delays in analysis and decision-making. Hence, in this chapter we focus on *online anomalous time series detection*, also known as real-time detection.

Several methods have been proposed in recent years to address this task. Chandola *et al.* [12] suggest a kNN based method which assigns anomaly score, equal to Euclidean distance to kth nearest neighbor in the data set, to each time series. Another distance measure based approach that can be used with kNN is dynamic time warping (DTW) proposed by Berndt and Clifford [3]. To address the time complexity of DTW, Keogh and Ratanamahatana [38] propose a lower bound based pruning technique to provide approximately linear time complexity. Fujimaki *et al.* [25] suggest Autoregressive (AR) approach which constructs a global AR model for all series and then calculates the anomaly score at time t as the gap between the observed value and the predicted value. Zhou *et al* [71] propose to apply the Back propagation Neural Network to the results of Local Outlier Factor(LOF), to analyze the outliers over data streams.

These methods are all based on single measure which may not achieve the best performance, as discussed in Chapter 4.

5.3 Online MUDIM Algorithms

5.3.1 A Naive Online MUDIM Algorithm (NMUDIM)

In Chapter 4, we proposed MUDIM, an anomaly detection method based on three distance measures:

- $dist_f(i,j)$: Standard deviation of differences (DiffStD) between ith and jth time series;
- $dist_t(i,j)$: Same trend (STrend) distance between ith and jth time series, which identifies the degree of synchronization of a series with another series; and
- $dist_s(i,j)$: SAXBAG distance between ith and jth time series, proposed by Lin [46], which captures patterns of a time series using a histogram of possible patterns.

All these three distance measures can be updated for incremental changes, as follows:

- **DIFFST**: The variance (square of the standard deviation) of differences between series i and j at time n can be calculated as:

$$\text{dist}_f(i,j) = \frac{n \times ssq(i,j) - (sqs(i,j))^2}{n \times (n-1)}, \quad (5.1)$$

$$\text{where } ssq(i,j) = \sum_{t=1}^{n}(x_i(t) - x_j(t))^2 \text{ and } sqs(i,j) = \sum_{t=1}^{n}(x_i(t) - x_j(t)) \quad (5.2)$$

  Clearly, the numerator in 5.1 can be updated for the $(n+1)$th observations by adding $(x_i(n+1) - x_j(n+1))^2$ and $(x_i(n+1) - x_j(n+1))$ to $ssq(i,j)$ and $sqs(i,j)$ respectively.

- **STrend** Let $x'_i(n) = x_i(n) - x_i(n-1)$. Then, by definition,

$$S_{i,j}(n) = \begin{cases} 1 & \text{if } x'_i(n) \cdot x'_j(n) > 0 \text{ or } x'_i(n) = x'_j(n) = 0 \\ 0 & \text{otherwise} \end{cases}$$

Consequently,

$$\text{dist}_t(i,j) = \frac{\sum_{t=2}^{n} S_{i,j}}{n-1}. \tag{5.3}$$

Therefore, to update it, for the $(n+1)$th observation we fix the numerator by adding the last trend $S_{i,j}(n+1)$ and accordingly modify the denominator as well.

- **SAXBAG** is based on SAX. It converts the data segment in the sliding window of size w to a single SAX word and then counts the frequency f_i of each word. When data at time $n+1$ is observed, a new SAX word will be generated based on $\{x_i(n+2-w), x_i(n+3-w), x_i(n+4-w), ..., x_i(n+1)\}$. The stored data set can be updated to account for the new SAX word.

- **Normalization and Assignment of Weights** Normalize $dist_f(i,j)$, $dist_t(i,j)$, and $dist_s(i,j)$ to $dist'_f(i,j)$, $dist'_t(i,j)$, and $dist'_s(i,j)$ respectively according to Step 2 in Algorithm 4.3.5. Then $\text{weight}'_\ell(i)$ be also calculated.

- Finally, anomaly score for ith series, $A_i = \sqrt{\sum_{\ell \in \{s,t,f\}} (\text{dist}'_\ell(i))^2 \times \text{weight}'_\ell(i)}$.

Based on the above equations, we first propose the "naive" online detection algorithm presented in Algorithm AlgNMUDIM . Anomaly scores, O_i's, for each time series can be plotted, for example see Figure 5.2.

Ignoring the length of a time series, the time complexity of NMUDIM is $O(m^2)$, because in MUDIM and NMUDIM we calculate distances $dist_l(i,j)$, $l = s,t,f$ for all $i \neq j$. In addition, the k nearest neighbor of series i are identified for each i. In the next subsection, we propose a method to reduce this complexity.

Algorithm AlgNMUDIM

Require: a positive integer k (number of nearest neighbor) , l (initial length for preprocessing) and data stream sets $\mathcal{D}$.

Ensure: anomaly score O for each series in $\mathcal{D}$.

1: $O = \emptyset$;
2: Initially length of all series in $\mathcal{D}$ is l.
3: **for** $i \in \mathcal{D}$ **do**
4: Calculate and store sax_i, f_i;
5: **for** $j \in \mathcal{D}$ **do**
6: Calculate $\text{dist}_s(i,j), \text{dist}_t(i,j), \text{dist}_f(i,j)$;
7: Calculate and store $S_{i,j}, ssq(i,j), sqs(i,j)$;
8: **end for**
9: **end for**
10: **while** new observations at time τ arrive **do**
11: **for** $i \in \mathcal{D}$ **do**
12: Update f_i, sax_i;
13: **for** $j \in \mathcal{D}$ **do**
14: Update $S_{i,j}, ssq(i,j), sqs(i,j)$ and $\text{dist}_s(i,j), \text{dist}_t(i,j), \text{dist}_f(i,j)$ based on equations 5.1 to 5.3;
15: **end for**
16: **end for**
17: **for** $i \in \mathcal{D}$ **do**
18: $A_i(k) =$ Equation in Step 5 in Algorithm 4.3.5
19: $O_i(\tau) = A_i(k)$;
20: **end for**
21: **end while**

Algorithm AlgOMUDIM

Require: a positive integer k (number of nearest neighbor) , l (initial length for preprocessing), a real number $\alpha \in [0..1]$(proportion of random sampling),$\beta \in [0..1]$(proportion of top used for threshold selection) and data stream sets $\mathcal{D}$.

Ensure: anomaly score O for each series in $\mathcal{D}$.

$O = \emptyset$.
Initially length of all series in $\mathcal{D}$ is l.
for $i \in \mathcal{D}$ **do**
  Calculate and store sax_i, f_i.
  for $j \in \mathcal{D}$ **do**
    Calculate $\text{dist}_{msm}(i,j)$ based on 5.4.
    Calculate and store $S_{i,j}, ssq(i,j), sqs(i,j)$.
  end for
end for
while new observations at time τ arrive **do**
  Find $\mathcal{S}$ which is randomly selected α proportion of the series from $\mathcal{D}$;
  for $i \in \mathcal{S}$ **do**
    for $j \in \mathcal{D}$ **do**
      Calculate $A_i(k)$ based on equation 5.5.
    end for
  end for
  Set threshold $h = A_\beta(k)$, where $A_\beta(k)$ is the top $(\beta \times 100)$th percentile of $A_i(k)$'s in descending order.
  Mark i as non-anomalous if its $A_i(k)$ is less than h.
  for $i \in \mathcal{D} - \mathcal{S}$ **do**
    for $j \in \mathcal{D}$ **do**
      Update $S_{i,j}, ssq(i,j), sqs(i,j)$ and $\text{dist}_{msm}(i,j)$ based on eqs 5.1 to 5.3
      Update $heap_i$ and $heap_j$ with $\text{dist}_{msm}(i,j)$ where $heap_i$ and $heap_j$ contain the k minimum dist_{msm} for series i,j
      if $average(heap_j) < h$ and $heap_j.size = k$ **then**
        mark j as non-anomalous;
      end if
      if $average(heap_i) < h$ and $heap_i.size = k$ **then**
        mark i as non-anomalous; **break**;
      end if
    end for
  end for
  for $i \in \mathcal{D}$ **do**
    if i is marked **then**
      $A_i(k) = 0$.
    else
      $A_i(k) = \frac{\sum_{j \in \mathcal{N}_k(i)} \text{dist}_{msm(i,j)}}{|\mathcal{N}_k(i)|}$.
    end if
    $O_i(\tau) = A_i(k)$
  end for
end while

5.3.2 Faster Online Detection of MUDIM (OMUDIM)

To speed up the process, instead of using rank based weighted distance in MUDIM 4.3.5, a simple weighted distance can also be used:

$$\text{dist}_{msm}(i,j) = \sqrt{\frac{\sum_{\ell\in\{s,t,f\}}(\text{dist}'_\ell(i,j))^2}{3}}. \tag{5.4}$$

Then, the anomaly score for the ith series , $A_i(k)$, can be defined as the average of $\text{dist}_{msm}(i,j)$ for i over its k nearest neighbor:

$$A_i(k) = \frac{\sum_{j\in\mathcal{N}_k(i)}\text{dist}_{msm}(i,j)}{|\mathcal{N}_k(i)|} \tag{5.5}$$

Now, select any k neighbors of i, and let $A'_i(k)$ denote the average $\text{dist}_{msm}(i,j)$ over them, then, obviously, average distance of k-nearest-neighbors of i must be less or equal to average distance of any k neighbors of i, so:

$$A_i(k) \leq A'_i(k). \tag{5.6}$$

In addition, if we can find a threshold λ such that $A_i(k) < \lambda$ implies i is not an anomalous series; then any $A'_j(k) < \lambda$ also implies j is not an anomalous series either; thus most of the non-anomalous series can be excluded from anomaly score calculations. This is the essence of OMUDIM. To find an estimate of the threshold, λ, we apply the following sampling procedure: we calculate $A_i(k)$'s for $i \in \mathcal{S}$, where $\mathcal{S}$ contains $\alpha \times 100$ percent of $\mathcal{D}$, randomly selected. Then λ is chosen to equal the value of $A_i(k)$ which is at the top $(\beta \times 100)$th percentile in descending order. Based on the above observations, we propose OMUDIM, a faster version of MUDIM, whose key steps are as follows:

1. Find λ as described above.

2. For $x_i \in \mathcal{D}-\mathcal{S}$, maintain a binary max heap consisting of $dist_{msm}(i,j)$ where j's are

selected k neighbors of i. If the average of these k neighbors is less than λ, series i is declared as non-anomalous. Else $dist_{msm}(i, j)$ is calculated for next selected value of j, and the heap is updated by keeping only the smallest k values of $dist_{msm}$. The anomalousness of series i is tested using the above criterion. This process stops if at any stage, the series is found to be non-anomalous or no j is left.

3. Calculate the anomaly scores of all potential anomalous series (found in Step2) and find the anomalous series, if any.

4. The above steps are repeated once new observations arrive.

The details of OMUDIM are given in Algorithm AlgOMUDIM. By applying these techniques, the time complexity of MUDIM is considerably reduced, as observed in experimental simulations.

5.4 Experiments

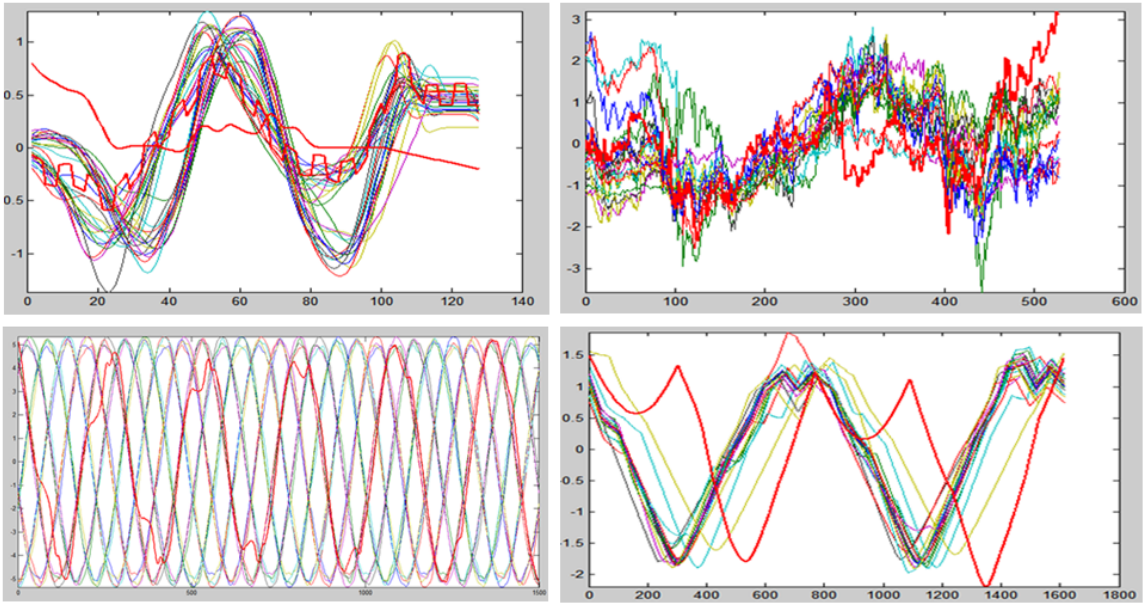

Fig. 5.1: **Experimental data sets** - Four data sets (from top to bottom, left to right) : SYN2, STOCKS, MOTOR, and SHAPE1. Anomalous series are marked as red.

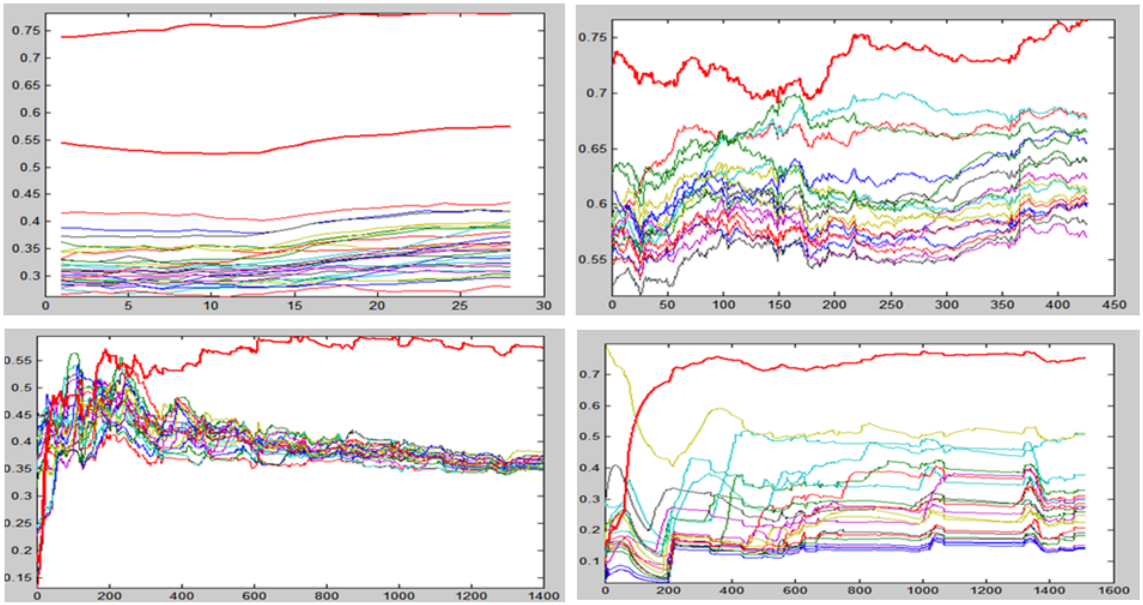

Fig. 5.2: **NMUDIM anomaly scores for each data set** - Plots of the online anomaly scores for each series at time 100+x in four data sets (from top to bottom, left to right) : SYN2, STOCKS, MOTOR, and SHAPE1. Anomalous series are marked as red.

We use eleven data sets, consisting of three synthetic data sets and eight modified real data sets introduced in Chapter 4, viz., SYN1, SYN2, SYN3, STOCKS, OPMZ, MOTOR, POWER, TEK140, TEK160, TEK170 and SHAPE1, to assess the effectiveness of the proposed algorithm. Key characteristics of the data sets are shown in Tables 4.3 and 4.4. Performance of our algorithms, measured in terms of capturing anomalous series, with existing online algorithms is presented in Table 5.1. We also calculate the running time of the algorithm for synthetic data streams. The comparisons of computational effort between the NMUDIM and OMUDIM are also shown in Table 5.2.

5.4.1 Results

In all experiments, the initialization is performed at $l = 100$th observations and the rest of the observations of the series are used to test the effectiveness of the proposed algorithm. The number of nearest neighbors is set at $k = 5$ for all data sets. All figures are obtained based on results of NMUDIM.

Table 5.1: Performance of all algorithms.
Numbers show the true outliers that are identified by algorithms within top p positions.

data sets	p =# of true outliers	Euclid	AR	DTW	NMUDIM	OMUDIM
SYN1	2	2	1	2	2	2
SYN2	2	2	2	2	2	2
SYN3	1	0	1	0	1	1
STOCKS	1	0	0	0	1	1
OPMZ	1	0	1	0	1	1
MOTOR	1	1	0	1	1	1
POWER	7	7	1	7	7	7
TEK140	1	1	0	1	1	1
TEK160	1	1	0	1	1	1
TEK170	1	1	0	1	1	1
SHAPE1	1	1	0	1	1	1

The data series in data sets are plotted in Figure 5.1 and the performance of the algorithm in Figure 5.2. As more data arrives and if more unusual patterns occur, the anomaly score increases and the gap between anomalous and normal series becomes larger. Normal series' anomaly scores converge if they are similar to each other.

We compare the performance of our algorithms with three other online detection algorithms based on (a) Euclidean distance, (b) Dynamic Time Warping (DTW), and (c) Autoregressive (AR) approach, proposed by[10], [41] and [25] respectively. The first two of these methods calculate a measure of anomalousness of a time series by (i) finding the k-nearest neighbors of the series, and (ii) using the average distance of these k-neighbors. The third method constructs a global AR model for all series and then measures the anomaly score at time t as the gap between the observed value and the predicted value.

To compare the performance of these algorithms, we simply compare the numbers of true anomalous series they detect once all observations have arrived,shown in Table 5.1.

It can be seen that our methods (NMUDIM and OMUDIM) perform very well for all 11 data sets; i.e., anomalous series are always in top p places. Other methods do well for some but not all sets. As seen in Chapter 4, this illustrates that no single "pure" metric is

sufficient for capturing multiple types of anomalies.

5.4.2 Time Complexity

To find the actual running complexity of the algorithm, six synthetic data sets were created. The algorithm was implemented by Matlab R2010b, and was run on a machine with Core i7, 6G memory, Windows 7 system.

The time complexity of the algorithm is $O(n \times m^2)$ because it updates stored data structures when new data arrive and then inter-time series distances are obtained for each pair of series. In addition, the k-nearest neighbors are obtained in order to calculate the anomaly scores.

As shown in all four figures, the anomalous series begins to differ from the rest of the group within as few as 100 additional observations. The hardest detection problem is in the upper right hand corner in Figures 5.1 and 5.2 consisting of stock prices; the anomalous series is difficult to be visualized. Even in this case, our algorithm succeeds in identifying the anomalous series. Experimental results describing computational effort are shown in Table 5.2. In those experiments, the parameters for OMUDIM were as follows: k is 1, α is 0.05 if number of series is less than 200, otherwise 0.1. β is 0.1. OMUDIM is about 60% faster than NMUDIM.

Table 5.2: Running time of NMUDIM and average computation workload comparison between NMUDIM and OMUDIM.

Running Time	Length	# of Series	NMUDIM (Seconds)	NUMDIM Workload	OMUDIM Workload	Ratio
Synthetic1	1500	20	3.90	190	63	33.2%
Synthetic2	1500	40	8.20	780	405	51.9%
Synthetic3	1500	80	18.28	3160	1138	36.0%
Synthetic4	1500	200	53.60	19900	6535	32.8%
Synthetic5	1500	400	152.69	79800	28387	35.6%
Synthetic6	1500	1000	706.64	499500	113304	22.7%

"Workload" represents the average number of comparisons performed in each iteration.

5.5 Concluding Remarks

We have proposed an algorithm for online anomaly detection of time series. This approach is efficient and detects anomalous series as soon as it begins to drift away from the other (non-anomalous) series, a substantial advantage over other anomaly detection algorithms for time series. Our approach can handle data uncertainty very well, and its online version only requires an initial length of data to start and doesn't require any training data sets. Compared with other methods, it requires less domain knowledge.

CHAPTER 6
ABNORMAL SUBSEQUENCE DETECTION IN A SINGLE SERIES

We addressed the problem of “which series is anomalous" in the preceding chapters, and need to answer another related question: “When"? In this chapter we address the detection of the subsequence of a single series which differs from the rest of the series, if anywhere. The chapter is organized as follows. First we make a precise statement of the problem in Section 6.1. We provide a quick review of existing approaches to address this problem in Section 6.2. Our new algorithm presents in Section 6.3, followed by results and conclusions in Section 6.4.

6.1 Problem Statement

This problem can be formulated as follows. Suppose $X = \{x(t)|1 \leq t \leq n\}$ is a time series where $x(t)$ represents the value of the series X at time t and n is the length of X. A subsequence of X is defined as $X_{p,w} = \{x(p), ..., x(p+w-1)|1 \leq p \leq n-w+1; 1 < w \leq n\}$. The goal is to find abnormal subsequence $X_{p,w}$, if any exists. More specifically, we must calculate the outlierness, $O(X_{p,w})$, of $X_{p,w}$. If $O(X_{p,w}) \geq O_{threshold}$; then $X_{p,w}$ is

declared to be abnormal. In this abnormal subsequence detection problem formulation, we leave users to determine value of $O_{threshold}$, and just address the computation of $O(X_{p,w})$ for each subsequence $X_{p,w}$.

6.2 Literature Review

Some researchers use model based methods to find abnormal subsequences. In this approach first a model is generated to predict the behavior of the time series. Using this model, the predicted values are calculated and compared with the observed values. The cumulative score of the differences is defined as the anomaly scores of observed data objects. These models include Regression by Fox [24], Rousseeuw and Leroy [55], Auto Regression by Fujimaki *et al.* [25], ARMA by Pincombe [52], ARIMA by Moayedi *et al.* [50], and Support Vector Regression by Ma *et al* [48]. These methods were mainly designed for individual outliers detection, not for abnormal subsequence detection, and some models are impacted largely by parameters or distributions of data sets. These deficiencies limit their practical application domains.

Keogh *et al.* [40] propose a method to convert a series to multiple shorter subsequences first, then use suffix tree and discretized subsequences to create an index structure for these subsequences, and finally compute anomaly scores by comparing trees generated by each subsequence. They also propose another technique for detecting irregular time series containing abnormal subsequences (they called it “discord") based on optimized dynamic time warping and SAX representation for time series [37, 41]. They introduce the concept of non-self match which means that it is meaningful to compare two subsequences only when they don’t have any overlapping data objects. They claim that after optimization, the computation’s speed of DTW is higher, but their approach simply detects only one abnormal subsequence or requires users to decide how many abnormal subsequences need to be detected before applying the algorithm. Wei *et al.* [63] propose a time series BITMAP method

based on comparison between the frequencies of SAX words of current data and past data. This approach some parameters whose appropriate values are hard to determine and may confuse users.

We propose an algorithm - a modification of MUDIM, which requires little or no domain knowledge and can detect multiple abnormal subsequences in one run. It also requires fewer parameters from users.

6.3 Proposed Method - Multiple Measure Based Abnormal Subsequence Detection Algorithm(MUASD)

The length of subsequence is a key factor used to decide the outlierness of a subsequence. In Figure 6.1, we show normal and abnormal subsequences; each subsequence represents power consumption on a weekly basis. Abnormal subsequences are plotted, along with the week that they represent. However, if subsequences represent daily power consumption, then it is quite possible that abnormal subsequences cannot be identified. Thus, anomalousness of a subsequence may depend on the length w of subsequence. Our algorithms rely on users to provide this length - w.

Detection of an abnormal subsequence of a given length w in a time series of length n appears to be a different problem from what we have studied, but actually it is quite similar to our previous work; consider each subsequence as a single time series and the longer time series is constituted of a set of shorter time series. Thus abnormal subsequence detection problem can be converted to anomalous time series detection problem discussed in Chapter 4. More precisely: given a time series X, and a set of subsequences, the set of extracted subsequences, $X_w = \{X_{p,w}; 1 \leq p \leq (n-w+1)\}$, consists of $X_{1,w} = \{x(1), x(2), \ldots, x(w)\}; X_{2,w} = \{x(2), x(3), \ldots, x(w+1)\}; \ldots; X_{n-w+1,w} = \{x(n-w+1), x(n-w+2), \ldots, x(n)\}$. One example is shown in Figure 6.2.

For abnormal subsequence detection, our algorithm uses the following criterion:**the dis-**

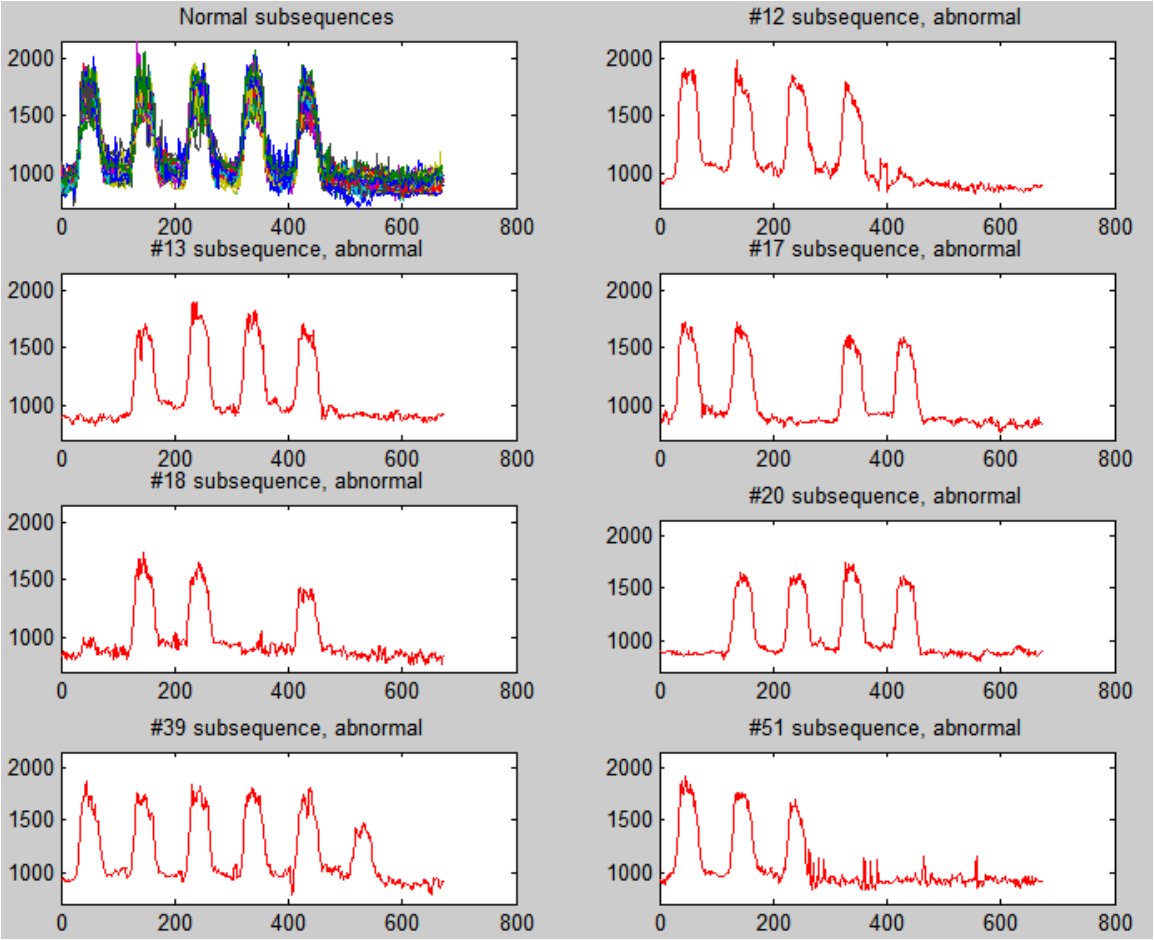

Fig. 6.1: **Examples for abnormal subsequences.** - Subsequences extracted from a long series of power consumption for a year; each subsequence represents power consumption for a week. (Top left) contains normal subsequences; (Others) contains abnormal subsequences along with which week they appear represent the power consumption during a week with special event or holidays.

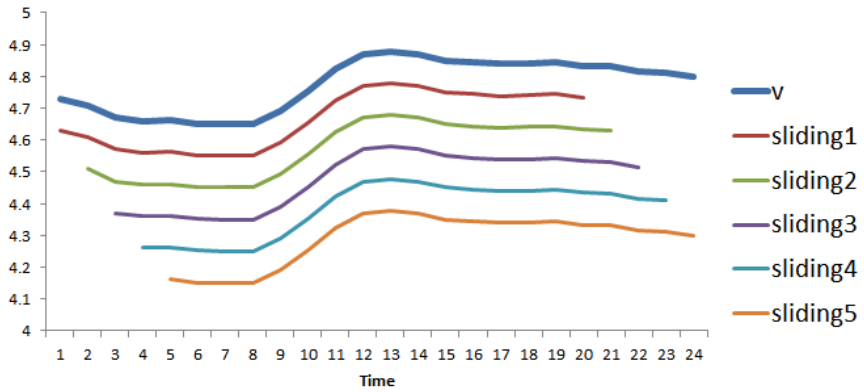

Fig. 6.2: **Illustration for sliding window concept.** - The top series is original long series; other series are subsequences generated by sliding window method. w is set to 20 in this case.

tance from an abnormal subsequence to its k nearest neighbors is significantly larger than a distance from a normal series to its k nearest neighbor. Based on this assumption, MUDIM algorithm introduced in Chapter 5 can be applied here to find the abnormal subsequences. However there are still some problems that need to be addressed.

Majority of data objects of any two consecutive series are common. Therefore, two consecutive subsequences will have a high chance to become the nearest neighbor of each other. Even the abnormal subsequence will find a nearest neighbor with very high similarity. Our assumption will fail and the algorithm will not be able to detect any abnormal subsequence.

The root cause of this issue is called the **self match**; a subsequence is compared with a clone of itself which may differ only by a small number of data objects. Keogh, *et al.*, suggest to use Non-Self Match [41] for a meaningful comparison. The basic idea is that one subsequence must be compared with another subsequence without any common data objects.

Now the algorithm is straightforward: just applying MUDIM to the set of series in X_w, which consists of all possible subsequences of X, each of length w. Next, find the k nearest neighbor subsequences for each subsequence $X_{p,w}$ in X_w using Non-Self Match. In other words, all $X_{p,w}$'s k nearest neighbor neighbors cannot have any common data objects with $X_{p,w}$. But $X_{p,w}$'s k nearest neighbor neighbors may have common data objects between each other.

MUDIM is a possible solution, but is not the optimal solution because of the reasons described below:

- MUDIM has to calculate the three measure for each subsequence, but for the set X_w, there is considerable redundant computation work involved because of data objects shared by different subsequences.

- To find the nearest neighbor, MUDIM has to compare a series with all other series. But in X_w, due to common data objects, there are some other methods that can help

to speed up the process by reducing the redundant work for common data objects between $X_{p,w}$.

- Frequencies of SAX words are used to calculate the distance between the series, but for this problem, other comparison method may be better.

In our algorithm, we propose two steps to speed up the process:

1. To find the nearest neighbor of a subsequence in X_w, we use Euclidean distance, and instead of k(>1) nearest neighbors we propose to use (1) the nearest neighbor only.

2. Comparing frequencies of SAX words of a subsequence $X_{p,w}$ with frequencies of same SAX words of the entire time series X is a more reasonable method.

We will introduce these two steps in detail in the following sections.

6.3.1 Finding Nearest Neighbor by Early Abandoning

In MUDIM, the k nearest neighbors are found based on MUDIM distances. But for abnormal subsequence detection, Euclidean distance is sufficient for finding a nearest neighbor for two reasons: 1) If we use Euclidean distance, the computational effort is reduced, especially when we exploit sliding window approach. 2) Because other measures are used, so deficiencies of Euclidean distance such as time-lagging can be overcome by using sliding window technique.

To search the nearest neighbor for a subsequence in a long time series, a “Reordering early abandoning" approach, used by Rakthanmanon *et al.* [54], can help us to further improve the speed. The idea of early abandoning is this: a distance which is the lowest distance within all distances calculated so far is called best-so-far distance. During future computation process, if current accumulated distance between each pair of data objects of two subsequences is already larger than best-so-far distance, then we can abandon this subsequence candidate since it cannot be the nearest neighbor for our interested subsequence.

“Reordering early abandoning" simply means that instead of comparing the pair of data objects of two subsequences from left to right, we can calculate the largest value objects first which may have higher chances to get large accumulated distances quickly. In Figure 6.3, an example of reordering early abandoning is shown.

Suppose the nearest neighbor for $X_{1,w}$ is $X_{p,w}$, then what is the possible nearest neighbor for $X_{2,w}$? Since majority objects in $X_{1,w}$ and$X_{2,w}$ are common, then $X_{p+1,w}$ would have a very high possibility to be the nearest neighbor for $X_{2,w}$. Thus, instead of searching nearest neighbor from the beginning, we can start at $X_{p+1,w}$, if $X_{p+1,w}$ is just the nearest neighbor to $X_{2,w}$, then the best-so-far distance for $X_{2,w}$ is the lowest and will speed up the processing by reordering early abandoning.

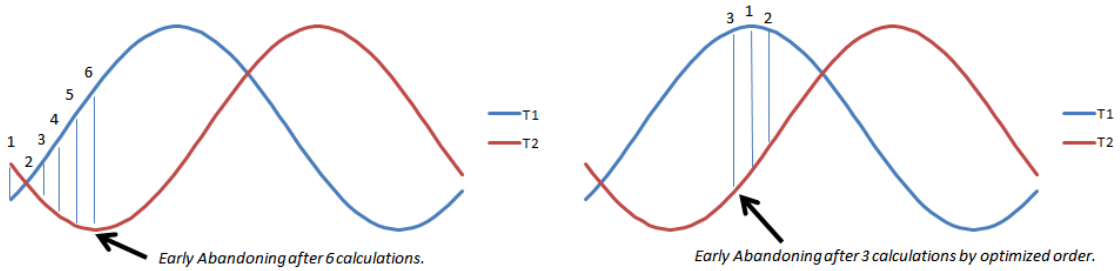

Fig. 6.3: **Illustration for reordering early abandoning.** - Left) After 6 object calculations, the accumulated distance between T1 and T2 is larger than best-so-far distance, stop here. Right) Calculate from the largest data objects first, we can stop only after 3 object calculations.

The function for finding the nearest neighbor by reordering early abandoning algorithm for MUASD can be implemented as shown in Algorithm FindNearestNeighbor.

6.3.2 Finding Abnormal Subsequence Based on Ratio of Frequencies (SAXFR)

Another important improvement is based on SAXBAG. In original MUDIM algorithm, we compare frequencies of SAX words of one series with frequencies of SAX words of another series. But actually there is a better way to do this. In the new method, k is not

Algorithm FindNearestNeighbor Finding the nearest neighbor subsequence for each subsequence using Non-Self Match.

Require: subsequence set X_w.

Ensure: locations ($locs$) and distances ($dists$) of nearest neighbor to every subsequence of length w in X.

```
1: N=|X_w|.
2: lastfound=0; locs = ∅; dists=∅.
3: for i = 1 : N do
4:    currSub = X_w[i].
5:    order = sortObjects(currSub, 'descending');
6:    best-so-far=infinity.
7:    nID = lastfound;
8:    nearestID=0;
9:    for j = 1 : N do
10:      nID = nID+1;
11:      if IsSelfMatch(i,nID) then
12:         continue; // Self match is ignored.
13:      end if
14:      neighborSub = X_w[nID].
15:      sums = 0;
16:      for p = 1 : w do
17:         sums = sums+ DIST(currSub, neighborSub, order, p);
18:         if sums>best-so-far then
19:            break;
20:         end if
21:      end for
22:      if sums<best-so-far then
23:         best-so-far=sums; nearestID = nID;
24:      end if
25:   end for
26:   locs_i = nearestID; // locations
27:   lastfound = nearestID;
28:   dists_i = best-so-far; // distances
29: end for
```

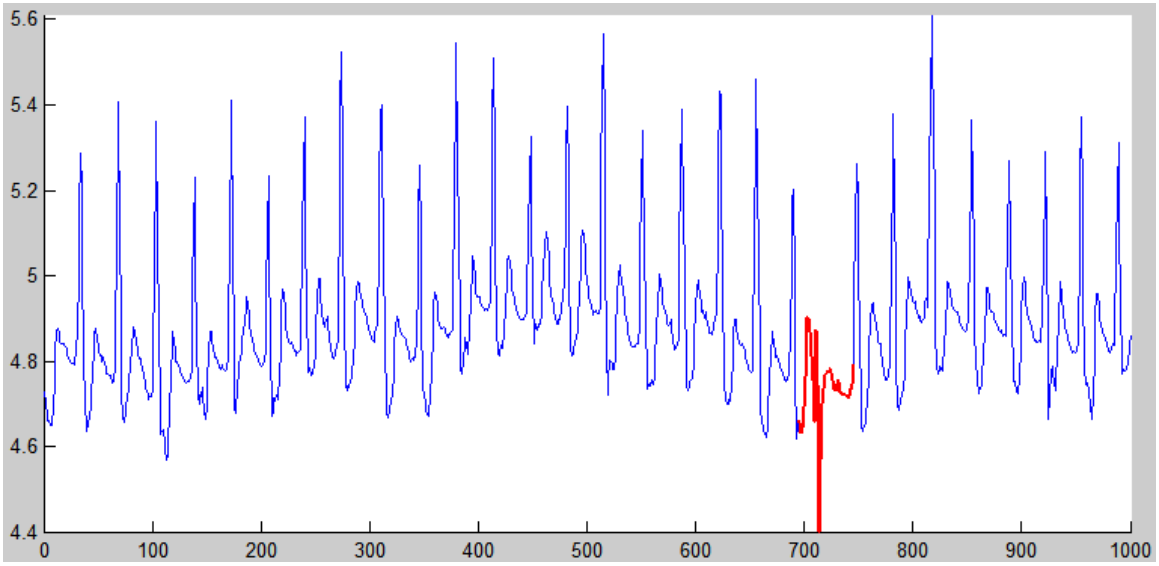

Fig. 6.4: **Frequency of abnormal subsequence is very low.** - Red bold line represents the abnormal subsequence. Abnormal subsequence only appear once in this series, but normal series appear more often.

needed any more and high accuracy can still be achieved. The basic idea is based on one assumption: the frequencies of abnormal subsequences are far lower than the frequencies of normal subsequences if length w is properly chosen. An example is shown in Figure 6.4. If we calculate the ratio between frequencies of SAX words generated from an abnormal subsequence with frequencies of these words in the entire series, then the ratio computed for an abnormal subsequence is much higher than the ratio computed for a normal series. For example, if three SAX words abc, aac, abd are generated from an abnormal subsequence, and these words only appear $2, 3, 2$ times within entire series, then the ratios of each word are $1/2, 1/3, 1/2$ respectively. Then an average ratio can be calculated for all non-zero ratios, in this case, it is $(0.50 + 0.33 + 0.50)/3 = 0.44$. For a normal series, this ratio should be much more lower. This new method is called SAX words based frequency ratio (SAXFR). To ensure SAX word is properly set to capture the outlierness of subsequence, the window size of SAX is set to $\frac{w}{2}$. If window size of each word in SAX is too short, then the shorter subsequence represented by each SAX word may not be anomalous. If window size is too long, then the number of SAX words obtained in each w length subsequence is

less and it may impact the results of the frequencies ratio approach.

The advantages of SAXFR are:

- Since the ratio is compared between a subsequence and the entire series, there is no need for nearest neighbor computation, and there is also no need for parameter k.
- Frequencies of entire series can be precomputed, and subsequence's frequencies can be updated incrementally, so the computation for ratios is very fast.

6.3.2.1 Length of Subsequence of SAXFR

We did some experiments to identify the relationship between size of subsequence and performance of SAXFR. In our experiments, three key parameters of SAX , sliding window size, number of symbols and length of SAX word, are set to half of sliding window size, 4 and 5 respectively. Sizes of subsequences :10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110 and 120, are tested for three data sets SYN0, ECG1 and ECG2. The minimum sizes of abnormal subsequences in these three data sets are 40, 40 and 100 respectively. The results are presented in Table 6.1 and Appendix C. The average of anomaly score of abnormal subsequences is always higher than that of normal subsequences.

In general, size of subsequence is not necessary equal to exact size of abnormal subsequence in order to achieve the best performance. In SYN0, the ratio of average anomaly score of abnormal subsequence over that of normal subsequence is larger than 3 when size of subsequence is from 10 to 50. As long as the size of subsequence increases, the ratio decreases. In ECG1, the ratio between average anomaly score of abnormal and normal subsequence is larger than 3 when size of subsequence is 20 and 30. In ECG2, the ratio is over 3 when size of subsequence is from 30 to 80.

According to the results of experiments, too small(like 10) or too large(like 120) size of subsequence both are not the best choices for SAXFR. Since if size is too small, then SAX word extracted from abnormal subsequence may be similar to the SAX word extracted from

Table 6.1: SAXFR performances versus length of subsequence

Dataset	w	Anomaly Scores		
		Average of abnormal $X_{p,w}$	Average of normal $X_{p,w}$	Ratio
SYN0	10	0.2733	0.0538	5.08
	20	0.3358	0.0653	5.14
	30	0.3448	0.0779	4.42
	40	0.3310	0.0764	4.33
	50	0.2775	0.0908	3.06
	60	0.2411	0.0978	2.47
	70	0.2231	0.1055	2.11
	80	0.2574	0.1215	2.12
	90	0.2260	0.1209	1.87
	100	0.1978	0.1246	1.59
	110	0.2087	0.1303	1.60
	120	0.1632	0.1213	1.34
ECG1	10	0.3761	0.1647	2.28
	20	0.5414	0.1309	4.14
	30	0.5176	0.1534	3.37
	40	0.3982	0.1509	2.64
	50	0.2769	0.1757	1.58
	60	0.2724	0.1821	1.50
	70	0.3133	0.1846	1.70
	80	0.3658	0.2141	1.71
	90	0.3446	0.2110	1.63
	100	0.3462	0.2402	1.44
	110	0.3546	0.2608	1.36
	120	0.3161	0.2510	1.26
ECG2	10	0.2209	0.1557	1.42
	20	0.2371	0.0976	2.43
	30	0.2480	0.0762	3.25
	40	0.2838	0.0852	3.33
	50	0.3003	0.0935	3.21
	60	0.3065	0.1000	3.06
	70	0.3395	0.1041	3.26
	80	0.3361	0.1077	3.12
	90	0.3460	0.1157	2.99
	100	0.3494	0.1178	2.96
	110	0.3571	0.1295	2.76
	120	0.3568	0.1381	2.58

normal subsequences. In other aspect, if size is too large, then the ratio of true abnormal subsequence in the subsequence is small, and it also causes the same problem.

The other factor that will impact the performance is the distribution of data, and more specifically is the similarity between normal subsequences. If normal subsequences are highly similar to each other such as SYN0, then the ratio of average anomaly score of abnormal subsequence over that of normal subsequence is usually larger.

6.3.3 Multiple Measure Based Abnormal Subsequence Detection Algorithm (MUASD)

Now, steps for MUASD are very straightforward:

1. Given a time series X, length of subsequence w, obtain X_w, the set of all possible subsequences of X using sliding window technique.
2. Get $locs$ and $dists$ using algorithm FindNearestNeighbor.
3. Calculate the frequencies ($FreqAll$) of each SAX word based on the entire series.
4. For each subsequence $X_{p,w} \in X_w$, calculate frequency $Freq_i$ for $X_{p,w}$, and compute the ratio $dist_r(i)$ of frequencies.
5. Calculate the sametrend distance $dist_t(i)$ between s_i and its nearest neighbor s_{locs_i}.
6. Normalize $dist_s, dist_r, dist_t$ (subtracting the mean and dividing the standard deviation).
7. Combine all distances and compute an anomaly score $O(i)$ for each ith subsequence.

In this algorithm we do not assign different weights to different features, as we did in MUDIM. The reason for this is that rank based method (and related methods) work better if the number of outliers is far less than the number of normal objects. But in the abnormal subsequence detection problem, the number of objects contained in an abnormal

subsequence is sometimes large, thus the number of subsequences containing common data objects with abnormal subsequence is also large. For instance, in ECG1 data set 6.7, the total number of data objects in abnormal subsequences is close to 10% of the entire series. A large number of anomalies will bias the results of rank based methods.

Algorithm MUASD Calculate the anomaly scores for each subsequence.

Require: a long time series X of length n, a length of subsequence w.
Ensure: anomaly scores $O(i)$ for ith subsequence, $1 \leq i \leq n - w + 1$.

1: X_w= ExtractSubsequences(X, w); {Subsequences are normalized to mean as zero.}
2: $win = \frac{w}{2}$; $word\_size = 5$; $symbol\_number = 4$; { Set parameters for SAX}
3: SW = GetSAXWords$(T, win, word_size, symbol_number)$; { Get sax words for X.}
4: $FreqAll$ = CalculateFrequencies(SW);
5: [$locs$, $dists$] = FindNearestNeighbor(X_w);
6: N=$|X_w|$.
7: $dist_e = dists$;
8: **for** $i = 1 : N$ **do**
9: $nid = locs_i$; //find the id of nearest neighbor.
10: $dist_t(i)$=CalculateSameTrendDist$(X_{i,w}, X_{nid,w})$; { Calculate sametrend distance between ith subsequence and nidth subsequence using equation }4.1.
11: SW_i = FindCurrentSAXWords(SW, i);
12: $Freq_i$=CalculateFrequencies$(SW_i, Freq_{i-1})$; {Update frequencies incrementally based on previous results if applicable, otherwise calculate from scratch}.
13: $dist_r(i) = NoneZeroMean(\frac{Freq_i}{FreqAll})$;
14: **end for**
15: $O(i) = \sqrt{dist_e(i)^2 + dist_t(i)^2 + dist_r(i)^2}$;

6.4 Experiments

6.4.1 Competing Algorithms

We selected some existing algorithms along with our algorithm MUASD to compare their performance. A length w of subsequence is given for every algorithm.

- (Euclidean distance method) Keogh *et al.* [41] use the Euclidean distance to nearest neighbor as anomaly score to find the most unusual subsequence. We apply their

method for every subsequence so that each subsequence can be assigned an anomaly score.

- (SAX Frequency Ratio) We also apply SAXFR separately to check its performance. The size of feature window is set to half of w. Alphabet size is set to 4 as suggest by Rakthanmanon *et al.* [54] and word size is set to 5. The parameters are chosen for the best performance.

- Model prediction based methods were used, e.g., Auto Regression by Fujimaki *et al.* [25], ARMA by Pincombe [52], ARIMA by Moayedi *et al.* [50]. Models are generated based on the entire series, and then the model is applied to predict the next observation using the model. The mean square's error is the anomaly score. Orders of parameter are chosen from 10 to 20, yielding the minimum mean square error.

The evaluation method here is very straight forward. We just compare the anomaly scores of abnormal subsequences (which are known to us) with those of normal subsequences. If anomaly scores of the abnormal subsequences are higher than those of normal subsequence, than the algorithm is effective; otherwise not.

6.4.2 Data sets

We use one synthetic data set and 7 real data sets. The synthetic data set contains several copies of 'normal' data and two abnormal subsequences, both exactly the same. Practical data sets are obtained from multiple application domains: two data sets are from medical domain, three are from electronic domain and two are from video surveillance domain. The details of data sets are given in Table 6.2. The length of w for each data set is chosen either by following the original author's suggestion (ECG) or by our observations.

Table 6.2: Time series data sets details.

Dataset	Source	Length	w	Domain
SYN0	Generated	966	40	Synthetic
ECG1	[63],[64]	1000	40	Medical
ECG2	[63],[64]	2160	40	Medical
TEK140	[41],[42]	5000	100	Electronic
TEK160	[41],[42]	5000	100	Electronic
TEK170	[41],[42]	5000	100	Electronic
VIDEOS1	[41],[42]	11251	200	Video Surveillance
VIDEOS2	[41],[42]	11251	200	Video Surveillance

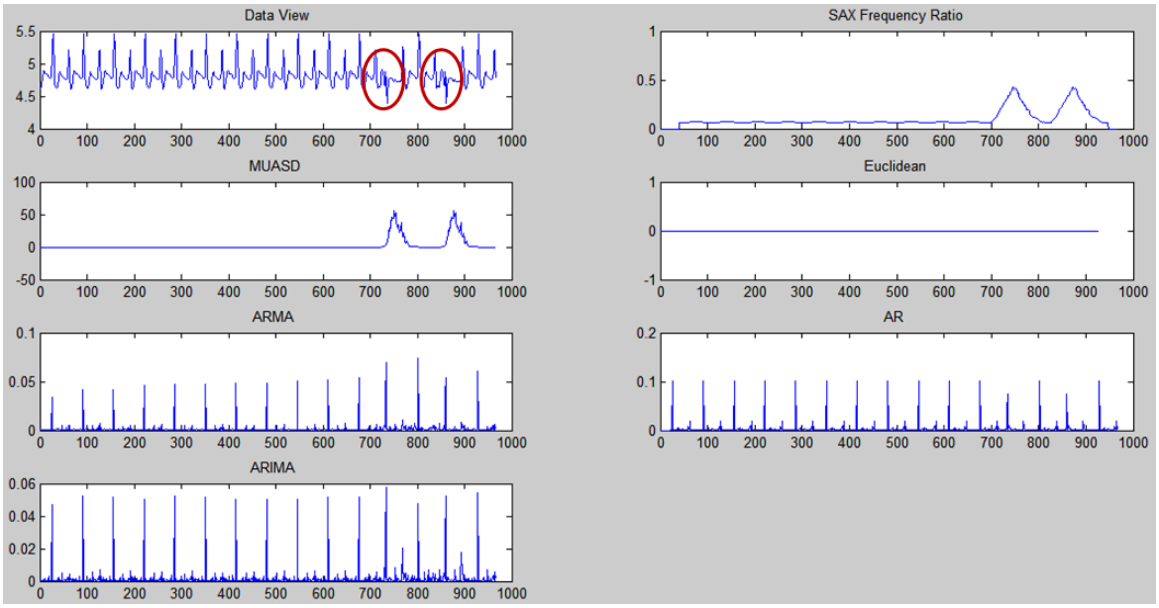

Fig. 6.5: **Experimental results for SYN0** - Red circles highlight abnormal subsequences. (Top Left) Plot of SYN0 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

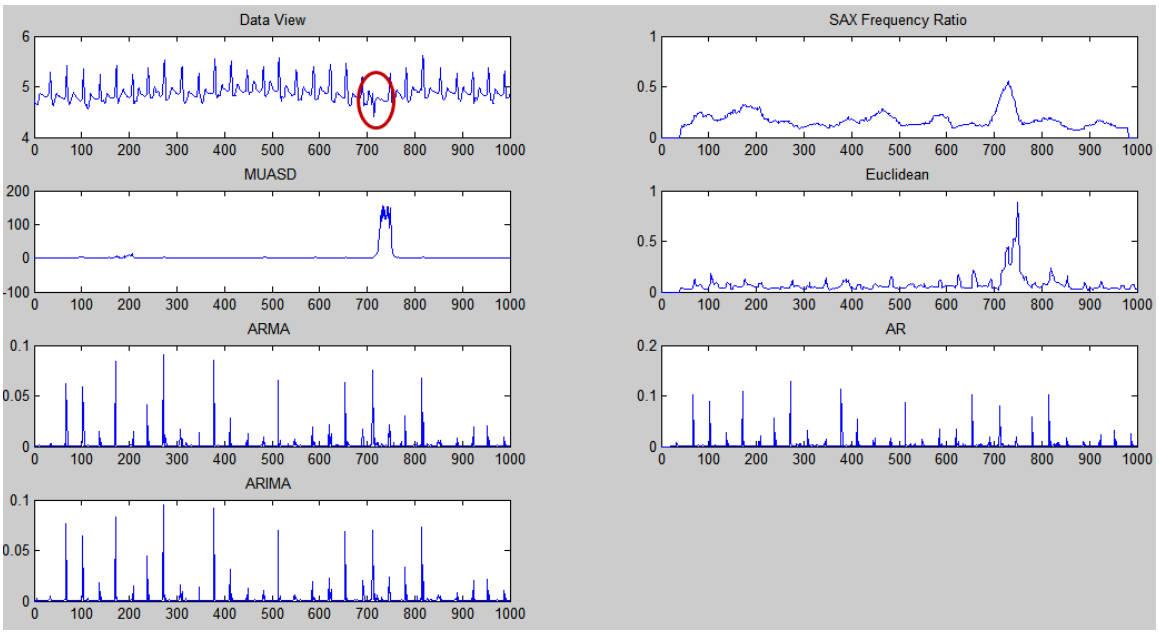

Fig. 6.6: **Experimental results for ECG1** - Red circle highlights abnormal subsequences. (Top Left) Plot of ECG1 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

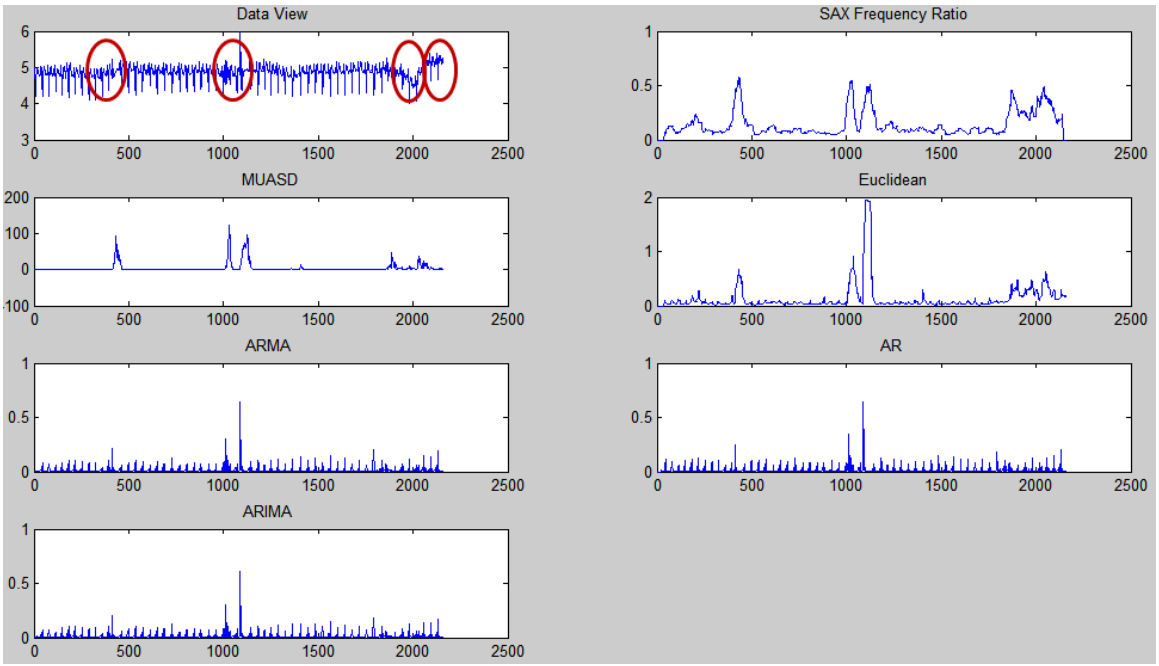

Fig. 6.7: **Experimental results for ECG2** - Red circle highlights abnormal subsequences. (Top Left) Plot of ECG2 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

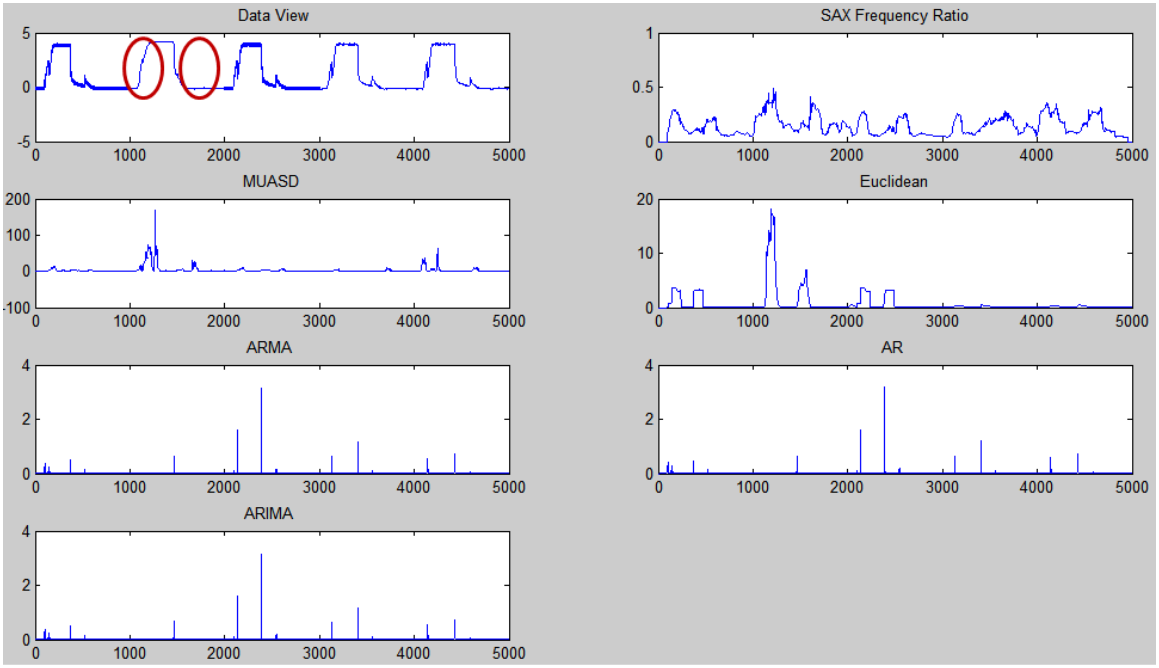

Fig. 6.8: **Experimental results for TEK140** - Red circle highlights abnormal subsequences. (Top Left) Plot of TEK140 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

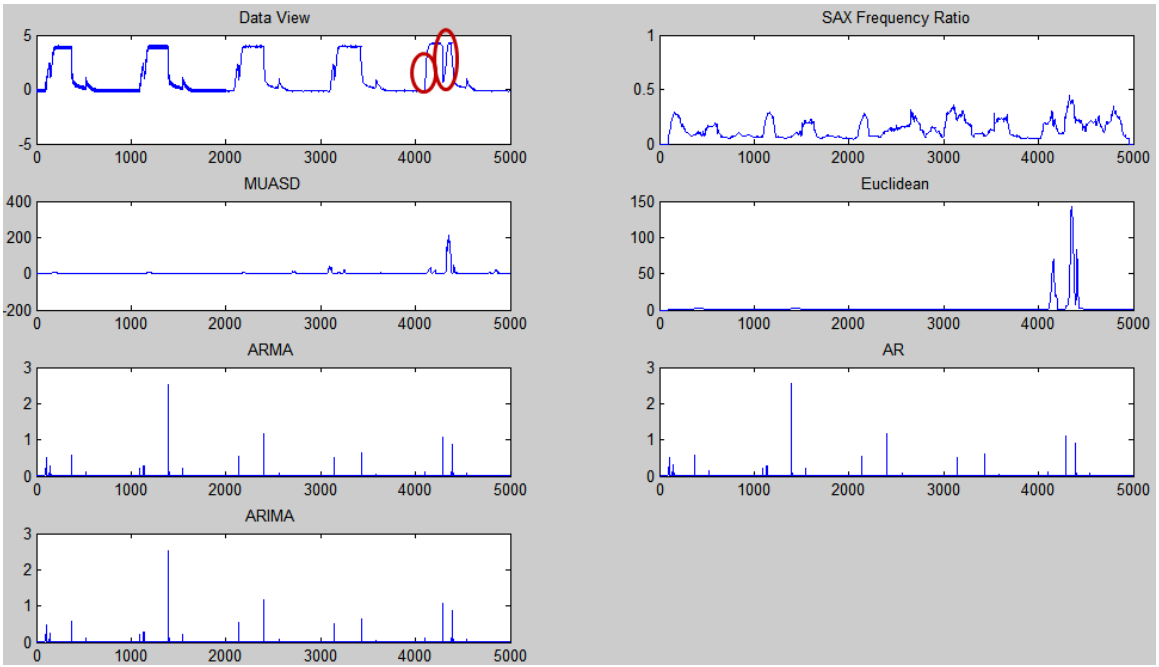

Fig. 6.9: **Experimental results for TEK160** - Red circle highlights abnormal subsequences. (Top Left) Plot of TEK160 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

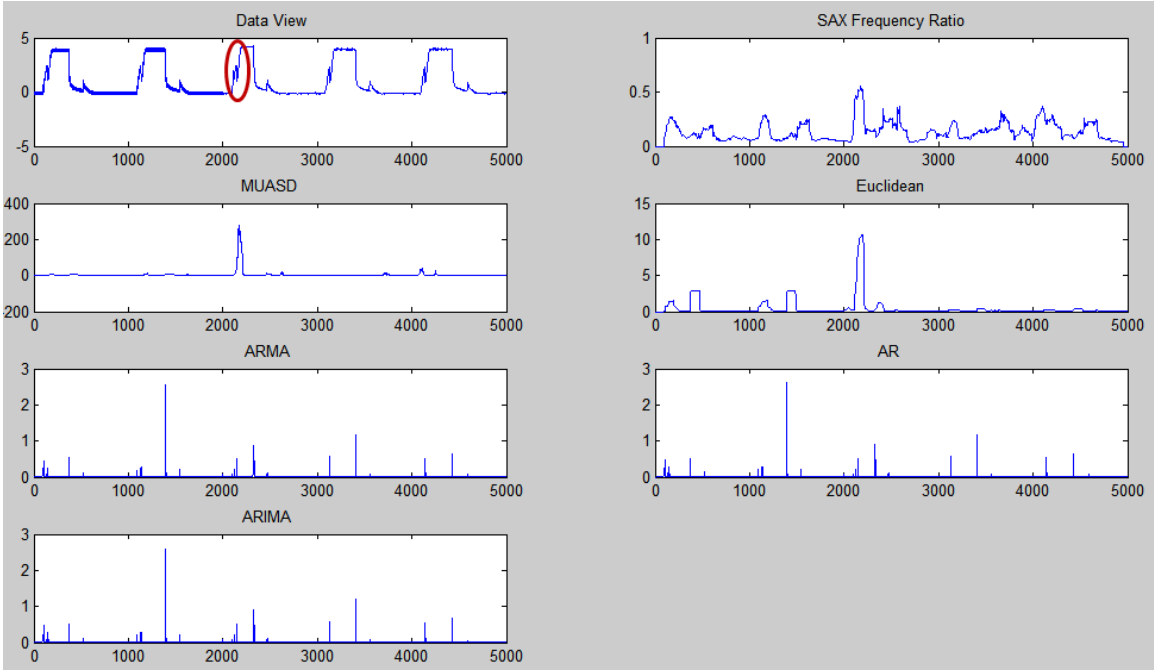

Fig. 6.10: **Experimental results for TEK170** - Red circle highlights abnormal subsequences. (Top Left) Plot of TEK170 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

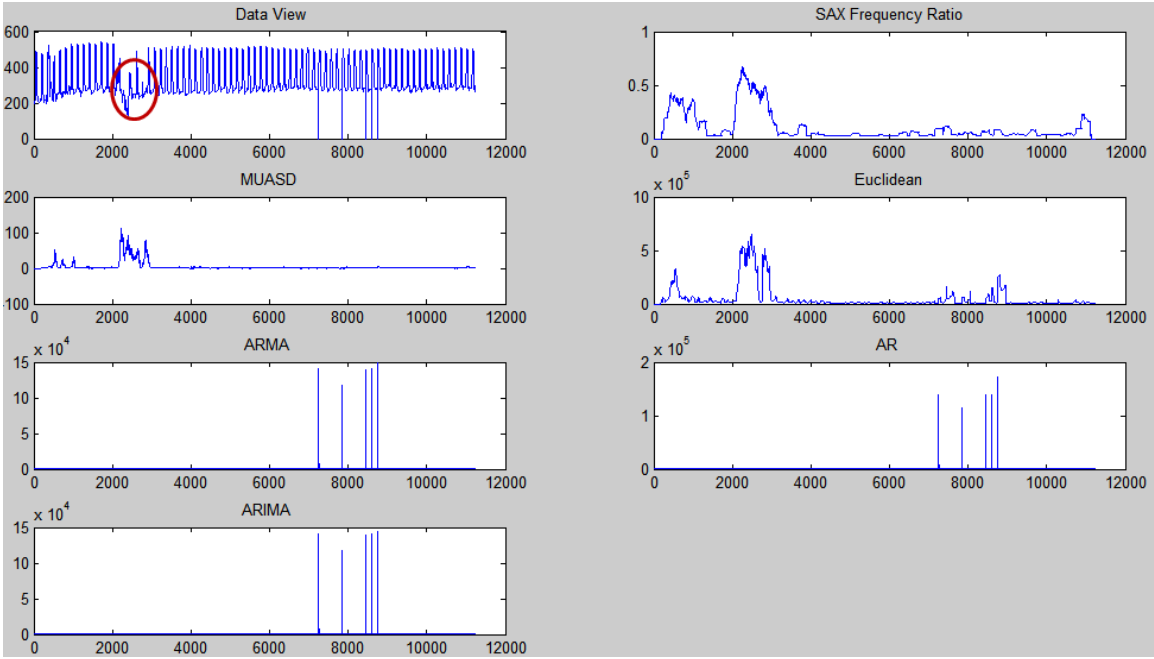

Fig. 6.11: **Experimental results for VIDEOS1** - Red circle highlights abnormal subsequences. (Top Left) Plot of VIDEOS1 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

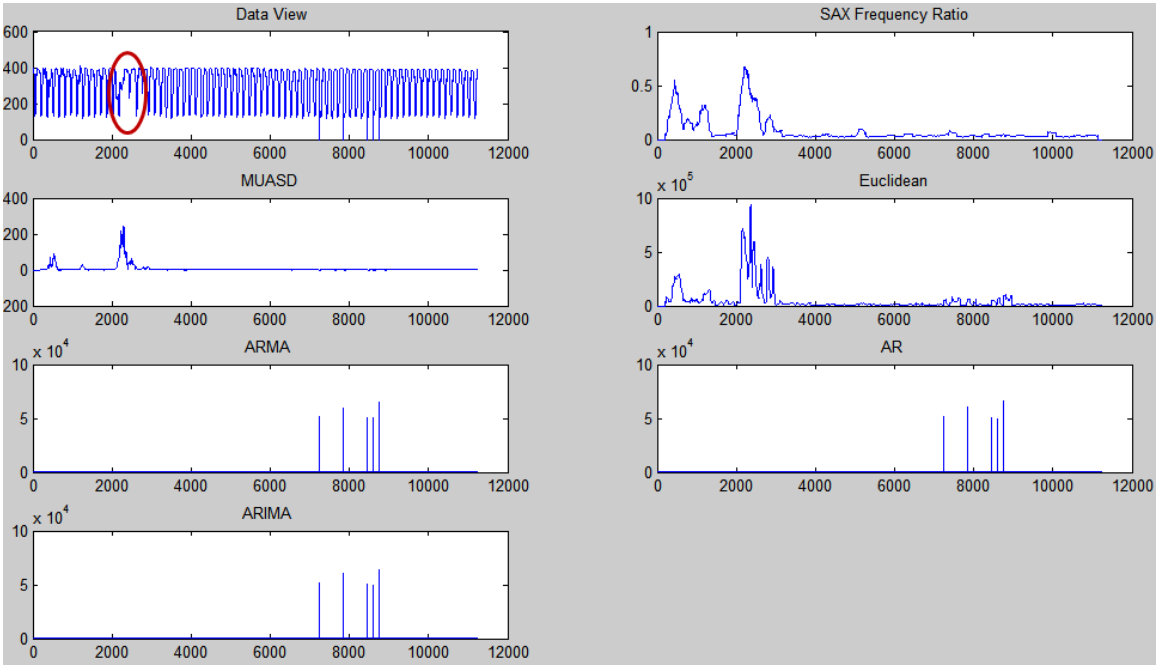

Fig. 6.12: **Experimental results for VIDEOS2** - Red circle highlights abnormal subsequences. (Top Left) Plot of VIDEOS2 time series; (Other) Results of 6 algorithms used in these comparisons. Y axis represents anomaly scores at time t. X axis shows time t.

6.4.3 Results

Experimental results are shown in Figures 6.5, 6.6, 6.7, 6.8, 6.9, 6.10, 6.11 and 6.12. Our observations are:

- Euclidean distance based method works well if abnormal subsequences are not similar to each other, see example in Figure 6.5. This approach results in false positives for some data sets such as TEK140 and TEK170.

- SAXFR method works well if normal subsequences appear more frequently. In TEK data sets, SAXFR doesn't work well since normal subsequences only appear four more times than abnormal subsequence and contain considerable noises.

- Model based methods work slightly well for data set ECG1 in Figure 6.6, but only one of three abnormal subsequences is detected. For other data sets, these do not work.

- MUASD usually works better for high frequency data sets. In low frequency data sets such as TEK, it may result in false positives.

6.5 Conclusions

In general, MUASD works better than other algorithms, even in the data sets that have exactly duplicated abnormal subsequences.

MUASD combines three important process: early abandoning searching for the nearest neighbor subsequence, frequencies ratio comparisons for each subsequence, and calculation for the same trend. Thus its time complexity is higher than Euclidean distance based method and SAXFR.

CHAPTER 7
CONCLUSIONS AND FUTURE WORKS

7.1 Conclusions

Anomaly or outlier detection problems are of considerable importance, arising frequently in diverse real-world applications such as finance and cyber-security. In the context of anomaly detection in financial institution (see Introduction) we argued that the collection of data associated with *all* users is likely to form multiple clusters with variable number of data points in each cluster, variable densities, etc; that is, no two clusters look alike, although they consist of 'normal' behavior. In such situations, to find anomalous observations, the ideal solution is to transform the data so that in the transformed space all data points have the same statistical distributions. In this dissertation we have attempted to achieve this goal via 'ranks' and 'modified ranks'. In using rank based approach (especially local to the object of interest) we manage to diminish the effect of inter-object distances; and in using 'modified-ranks' we diminish the effect of the size of the local cluster(s).

Several algorithms have been formulated for anomaly detection, usually based on a problem-dependent heuristic or distance metric. To achieve the better performance of anomaly detection, we transform the original data set into a transformed space such that all normal data objects are more closer to each other while anomalous objects are far away

from them.

In this dissertation we have addressed three important outlier detection problems: detection of 1) point outliers in a given data set; 2) anomalous time series in a given time series data set; and 3) abnormal subsequence in a single series. We propose anomaly detection algorithms that exploit the notion of “rank," expressing relative outlierness of different points in the relevant space. Ranks also exploit asymmetry in nearest neighbor relation between points: **a data point is “more anomalous" if it is not the nearest neighbor of its nearest neighbors**. Although rank is computed using distance, it is a more robust and higher level abstraction that is particularly helpful in problems characterized by significant variations of data point density, where distance alone is inadequate. Our anomaly detection algorithms have many advantages compared with other existing approaches:

1. No apriori training set or reference set for normal behavior is needed.
2. Anomalies can be effectively detected even if the distribution of data is unknown.
3. It can be applied or modified for outlier detection in different domains.
4. Number of user-provided parameters is small.

1. Point outlier detection For point outliers detection, we have proposed novel algorithms based on ranks, that captures the anomalousness of an object more precisely in most cases. Moreover it can be applied in multiple domains arising the real world situations. In conjunction with clustering algorithm these rank based algorithms can be further improved; by identifying a small subset that do not belong to any cluster as “potential" outliers. Extensive evaluations on synthetic and real datasets have demonstrated that the overall performance of each of our new algorithms is significantly better than previously known algorithms. Among the new variants of algorithms, ODMRW performed best, perhaps due to the reason that it takes into account the robustness of rank methods and distance which is considered to be the primary measure of normal versus anomalous objects.

2. Anomalous time series detection To detect anomalous time series, we propose an algorithm that uses multiple distance measures; thus it performs better than single distance measure based methods because it can capture a variety of anomalous series with less domain knowledge. The selection of three measures is based on several heuristics. Rank based algorithms are applied to adjust the weights of different measures automatically. The weight assigned to a measure is proportional to its effectiveness in detecting anomalousness of a series in the data set, as described in Chapter 2.

The outlierness of a series can be incrementally calculated, which allows its application for online detection. We propose two versions of online detection algorithms based on multiple distance measures. These approaches are efficient and detect anomalous series as soon as it begins to drift away from the other (non-anomalous) series, a substantial advantage over other anomaly detection algorithms for time series. Our approaches can handle data uncertainty very well, and only requires an initial length of data to start and don't require any training data sets. Compared with other methods, our algorithms require less domain knowledge.

3. Abnormal subsequence detection For abnormal subsequence detection, we propose a novel algorithm based on nearest neighbor and multiple distance measures. Our approach only requires one parameter which is a user-provided length of subsequence that potentially contains abnormality and uses the entire time series. The algorithm achieves higher accuracy than other known methods, even in the data sets that have more than one copy of abnormal subsequences; other algorithms may fail in this situation.

7.2 Future Work

Following applications of rank based anomaly detection of a point or a time series are planned for future research.

7.2.1 Rank Based Time Series Anomaly Detection

Presently, we transform a given series and evaluate its anomalousness in transformed space. Alternatively, we can consider $\mathcal{D}$ as a collection of column vectors, where each vector is a collection of observations of all time series at a given time point. That is $\mathcal{D} = \{v_1, \ldots, v_n\}$ where, for example , $v_1^T = \{x_1(1), x_2(1), \ldots, x_m(1)\}$. Now we can find anomalies in each vector and then analyze the number of anomalies associated with each time series. This potential alternative approach to detect anomalous time series in $\mathcal{D}$ will be explored and compared with existing methods.

7.2.2 Clustering Based Anomalous Time Series Detection Approach

Another possible research direction is implementation of other clustering algorithms. Currently, only $\mathcal{NC}(\ell, m^*)$ clustering algorithm is explored, but other clustering algorithms in conjunction with ranks are also worth investigating. The clustering algorithm, $\mathcal{NC}(\ell, m^*)$ has been exploited for anomaly detection only. Another potential application is to use its center for other applications, for example in many applications users want to find clusters of similar time series and use the center for trading applications.

7.2.3 Time Complexity of Online Detection Algorithms

To reduce the time complexity of the online detection algorithm ihe other possible research direction is to take advantages of as much information as possible. In NMUDIM method, new observations arrival at t will cause each series in $\mathcal{D}$ to find its kNN by comparing with all other series in $\mathcal{D}$. When the number of series m is large, this process is computationally expensive. But actually, it can be avoided by making use of its k nearest neighbors at $t-1$, which are already known. If distances of these neighbors have not changed significantly, then x_i is more likely a normal series; otherwise, old method may be needed to find new

kNN for x_i. In most cases, normal series continue to be normal, thus the computation cost will be reduced from $n \times m^2$ to $n \times m \times k$ which is significant improvement when m is very large.

7.2.4 Anomalous Multivariate Time Series Detection

Anomalous multivariate time series detection is more complicated than univariate time series detection. To extend our algorithms to multivariate time series, we can use the following possible methods:

- Multivariate time series data set can be treated as multiple univariate time series data set. Then our algorithms can be applied for each univariate time series data set separately and then all results of each data set can be merged together by rank based weighted method. Or
- Select other multivariate-adaptable features into our algorithms such as covariance matrix.

Our rank based approach can be used in multivariate time series to adjust the weights not only in different features but also in different variables.

7.2.5 Time Series Data Issues

Two major 'real-world' properties of two series have been overlooked in our work. They consist of

- Lack of alignment.
- Missing data.

We will explore how to account for these problems in our algorithms.

REFERENCES

[1] J. Abfalg, H.-P. Kriegel, P. Kroger, P. Kunath, A. Pryakhin, and M. Renz, "Similarity search on time series based on threshold queries," *In Proceeding EDBT'06 Proceedings of the 10th international conference on Advances in Database Technology,Springer-Verlag Berlin, Heidelberg*, pp. 276–294, 2006.

[2] R. Baeza-Yates and B. Ribeiro-Neto, *Modern information retrieval.* Addison-Wesley Longman Publishing Co., Inc. Boston, 1999.

[3] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *In: AAAI Working Notes of the Knowledge Discovery in Databases Workshop*, pp. 359–370, 1994.

[4] G. Bonanno, F. Lillo, and R. N. Mantegna, "High-frequency cross-correlation in a set of stocks," *Quantitative Finance, 1,Jan 2001*, pp. 96–104, 2001.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *In Proceedings of the ACM SIGMOD International Conference on Management of Data. ACM Press*, pp. 93–104, 2000.

[6] Y. Bu, T. wing Leung, A. W. chee Fu, E. Keogh, J. Pei, and S. Meshkin, "Wat: Finding top-k discords in time series database," in *In Proceedings of 7th SIAM International Conference on Data Mining*, 2007.

[7] H. Cao, G. Si, Y. Zhang, and L. Jia, "Enhancing effectiveness of density-based outlier mining scheme with density-similarity-neighbor-based outlier factor," *Expert Systems with Applications: An International Journal*, vol. 37, no. 12, December 2010.

[8] K. P. Chan and A. W. C. Fu, "Efficient time series matching by wavelets," *In Proceeding ICDE '99 Proceedings of the 15th International Conference on Data Engineering, Sydney, Austrialia, March 23-26, 1999*, p. 126, 1999.

[9] P. K. Chan and M. V. Mahoney, "Modeling multiple time series for anomaly detection," *15th IEEE International Conf on Data Mining*, pp. 90–97, 2005.

[10] D. C. V. Chandola and V. Kumar, "Detecting anomalies in a time series database," *Technical Report 09-004*, 2009.

[11] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, p. ARTICLE 15, July 2009.

[12] V. Chandola, V. Mithal, and V. Kumar, "A comparative evaluation of anomaly detection techniques for sequence data," in *2008. ICDM '08. Eighth IEEE International Conference on Data Mining*, 2008, pp. 743 – 748.

[13] A. W. chee Fu, O. T. wing Leung, E. Keogh, and J. Lin, "Finding time series discords based on haar transform," in *In Proceeding of the 2nd International Conference on Advanced Data Mining and Applications*. Springer Verlag, 2006, pp. 31–41.

[14] L. Chen and R. Ng, "On the marriage of lp-norms and edit distance," in *In VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, 2004, pp. 792–803.

[15] Y. Chen, M. A. Nascimento, B. Chin, O. Anthony, and K. H. Tung, "Spade: On shape-based pattern detection in streaming time series," *Data Engineering, 2007. ICDE*

2007. IEEE 23rd International Conference, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007., pp. 786 – 795, 2007.

[16] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proceedings of the VLDB Endowment*, pp. 1542–1552, August 2008.

[17] L. Ertoz, M. Steinbach, and V. Kumar, "Finding topics in collections of documents: A shared nearest neighbor approach," in *In Clustering and Information Retrieval*, 2003, pp. 83–104.

[18] M. Ester, H. peter Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 226–231.

[19] A. Fabrizio and P. Clara, "Outlier mining in large high-dimensional data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 2, pp. 203–215, 2005.

[20] A. Fabrizio, B. Stefano, and P. Clara, "Distance-based detection and prediction of outliers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 2, pp. 145–160, 2006.

[21] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos., "Fast subsequence matching in time-series databases," *In Proceedings of the 1994 ACM SIGMOD international conference on Management of data, New York, NY, USA*, pp. 419 – 429, 1994.

[22] J. Feng, Y. Sui, and C. Cao, "Some issues about outlier detection in rough set theory," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4680–4687, 2009.

[23] Z. Ferdousi and A. Maeda, "Unsupervised outlier detection in time series data," *ICDE Workshops*, p. 121, 2006.

[24] A. J. Fox, "Outliers in time series," in *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 34, no. 3, 1972, pp. 350–363.

[25] R. Fujimaki, T. Yairi, and K. Machida, "An anomaly detection method for spacecraft using relevance vector," in *Learning, The Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. Springer, 2005, pp. 785–790.

[26] Fusion Media Limited, "Commodity prices," Website, 2012, http://www.forexpros.com/commodities/real-time-futures.

[27] S. Guha, R. Rastogi, and K. Shim, "An efficient clustering algorithm for large databases," *In Proceedings of the 1998 ACM SIGMOD international conference on management of data, Seattle, Washington, USA*, pp. 73–84, 1998.

[28] ——, "Rock: A robust clustering algorithm for categorical attributes," in *15th International Conference on Data Engineering, 1999. Proceedings.,*, 2000, pp. 512–521.

[29] D. M. Hawkins, *Identification of Outliers*, 1st ed. Chapman and Hall, London, 1980.

[30] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, "Inlier-based outlier detection via direct density ratio estimation," *In Proceedings of the 2008 Eighth IEEE international conference on data mining, Washington DC, USA*, 2008.

[31] H. Huang, K. Mehrotra, and C. K. Mohan, "Rank-based outlier detection," *Journal of Statistical Computation and Simulation*, vol. 82, pp. 1–14, 2011.

[32] ——, "Algorithms for detecting outliers via clustering and ranks," in *IEA/AIE'12, The 25th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Dalian, China*, 2012, pp. 20–29.

[33] ——, "An online anomalous time series detection algorithm for univariate data streams," in *IEA/AIE'13, The 26th International Conference on Industrial, Engineer-*

ing and Other Applications of Applied Intelligent Systems, Amsterdam, Netherlands, 2013.

[34] H. Huang, K. Mehrotra, and C. Mohan, "Detection of anomalous time series based on multiple distance measures," *28th International Conference on Computers and Their Applications (CATA-2013), Honolulu, Hawaii, USA*, 2013.

[35] C. Jeffery, "Synthetic lightning emp data," 2005. [Online]. Available: http://nis-www.lanl.gov/~eads/datasets/emp

[36] W. Jin, A. K. H. Tung, J. Han, and W. Wang, "Ranking outliers using symmetric neighborhood relationship," *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 577–593, 2006.

[37] E. Keogh, J. Lin, S. H. Lee, and H. V. Herle, "Finding the most unusual time series subsequence: algorithms and applications," *Knowledge and Information Systems*, vol. 11, no. 1, pp. 1–27, 2006.

[38] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Know l. Inf. Syst.*, vol. 3, pp. 358–386, 2005.

[39] E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei, and C. Ratanamahatana, "The UCR time series classification/clustering," Website, 2011, http://www.cs.ucr.edu/~eamonn/time_series_data/.

[40] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: A survey and empirical demonstration," in *In the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 102–111.

[41] E. Keogh and J. Lin, "Hot SAX: Efficiently finding the most unusual time series subsequence," *In Proc. of the 5th IEEE International Conf on Data Mining (ICDM 2005).Houston, Texas, Nov 27-30, 2005.*, pp. 226 – 233, 2005.

[42] ——, "Hot sax: Efficiently finding the most unusual time series subsequence," Website, 2005, http://www.cs.ucr.edu/~eamonn/discords/.

[43] E. M. Knorr and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," *Proceeding VLDB '98 Proceedings of the 24rd International Conference on Very Large Data Bases*, pp. 392–403, 1998.

[44] M. Leng, X. Lai, G. Tan, and X. Xu, "Time series representation for anomaly detection," *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on 8-11 Aug. 2009*, pp. 628–632, 2009.

[45] J. Lin, E. Keogh, A. Fu, and H. V. Herle, "Approximations to magic: Finding unusual medical time series," *IEEE Symposium on CBMS*, pp. 329–334, 2005.

[46] J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation," *Journal of Intelligent Information Systems (1 April 2012)*, pp. 1–29, 2012.

[47] Z. Liu, J. X. Yu, L. Chen, and D. Wu, "Detection of shape anomalies: A probabilistic approach using hidden markov models," *ICDE*, pp. 1325–1327, 2008.

[48] J. Ma, S. Perkins, and A. B. Corp, "Online novelty detection on temporal sequences," 2003, pp. 613–618.

[49] X. Meng and Z. Chen, "On user-oriented measurements of effectiveness of web information retrieval systems," *In Proceeding of the 2004 international conference on internet computing.*, pp. 527–533, 2004.

[50] H. Z. Moayedi and M. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *ISIT*, vol. 4, 2008, pp. 1–6.

[51] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast distributed outlier detection in mixed-attribute data sets." *Data Mining Knowledge Discovery*, vol. 12, pp. 203–228, 2006.

[52] B. Pincombe, "Anomaly detection in time series of graphs using arma processes," in *ASOR BULLETIN*, vol. 24, no. 4, 2005, pp. 2–10.

[53] P. Protopapas, J. M. Giammarco, L. Faccioli, M. F. Struble, R. Dave, and C. Alcock, "Finding outlier light curves in catalogues of periodic variable stars," *Monthly Notices of the Royal Astronomical Society*, vol. 369, no. 2, pp. 677–696, 2006.

[54] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceeding KDD '12 Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 262–270.

[55] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*. John Wiley and Sons Inc.

[56] M. S. Sadik and L. Gruenwald, "DBOD-DS: distance based outlier detection for data streams," *Proceeding DEXA'10 Proceedings of the 21st international conference on Database and expert systems applications: Part I*, pp. 122–136, 2010.

[57] G. Salton, *Automated text processing: The transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc. Boston, 1998.

[58] J. Tang, Z. Chen, A. W. Fu, and D. W. Cheung, "Capabilities of outlier detection schemes in large datasets, framework and methodologies." *Knowledge and Information Systems*, vol. 11, no. 1, pp. 45–84, 2006.

[59] J. Tang, Z. Chen, A. W. chee Fu, and D. W. Cheung, "Enhancing effectiveness of outlier detections for low density patterns," *In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 535–548, 2002.

[60] Y. Tao and D. Pi, "Unifying density-based clustering and outlier detection," *2009 Second International Workshop on Knowledge Discovery and Data Mining, Paris, France*, pp. 644–647, 2009.

[61] D. Toshniwal and S. Yadav, "Adaptive outlier detection in streaming time series," *Proceedings of International Conference on Asia Agriculture and Animal(ICAAA), Hong Kong*, pp. 186–192, 2011.

[62] L. Wei, E. Keogh, and X. Xi, "Saxually explicit images: Finding unusual shapes," in *In proceedings of the 2006 IEEE International Conference on Data Mining. Hong Kong. Dec*, 2006, pp. 18–22.

[63] L. Wei, N. Kumar, V. Lolla, E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Assumption-free anomaly detection in time series," *Proceedings of the 17th International Conference on Scientific and Statistical Database Management (SSDBMą́r05*, pp. 237–242, 2005.

[64] ——, "Assumption-free anomaly detection in time series," Website, 2005, http://alumni.cs.ucr.edu/~wli/SSDBM05/.

[65] Y. Xie and S. Tang, "Online anomaly detection based on web usage mining," in *Proceeding IPDPSW '12 Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops and PhD Forum*, 2012, pp. 1177–1182.

[66] Yahoo Inc, "Stocks price dataset," Website, 2012, http://finance.yahoo.com/q/hp?s=WMT+Historical+Prices.

[67] K. Yamanishi and J. Takeuchi, "A unifying framework for detecting outliers and change points from non-stationary time series data," *In Proc. of the Eighth ACM SIGKDD, ACM*, pp. 676–681, 2002.

[68] D. Yankov, E. Keogh, and U. Rebbapragada, "Disk aware discord discovery: Finding unusual time series in terabyte sized datasets," in *In Proc. IEEE Int. Conf. Data Mining*, 2007, pp. 381–390.

[69] D. Yu, G. Sheikholeslami, and A. Zhang, "Findout: Finding out outliers in large datasets," *Knowledge and Information Systems*, vol. 4, no. 4, pp. 387–412, 2002.

[70] Y. Zhang, S. Yang, and Y. Wang, "LDBOD: A novel local distribution based outlier detector," *Pattern Recognition Letters*, vol. 29, no. 7, pp. 967–976, 2008.

[71] J. Zhou, Y. Fu, Y. Wu, H. Xia, Y. Fang, and H. Lu, "Anomaly detection over concept drifting data streams," *Journal of Computational Information Systems*, vol. 5, pp. 1697–1703, 2009.

VITA

NAME OF AUTHOR: Huaming Huang

PLACE OF BIRTH: Sanming, Fujian, P.R. China

DATE OF BIRTH: Dec 29, 1977

EMAIL: hhuang13@syr.edu

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

Bachelor Degree, 2000, Xiamen University, P.R. China

Master Degree, 2009, Syracuse University, U.S.A

ASSOCIATION MEMBERSHIPS:

Member of International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, 2011, 2013.

PROFESSIONAL EXPERIENCE:

- Sales forecasting for marketing department, DELL, 2003-2005. Implementing algorithms to do sales forecasting for marketing department.
- Fixed income analysis and anomaly detection, Syracuse University, 2011-2012. Developing and implementing the algorithms for detecting anomaly observation on daily basis.

- Text mining for name recognition, Syracuse University, 2013. Developing and implementing the algorithms for company names recognition in large data set.

RESEARCH INTERESTS

I have broad interests in anomaly detection, such as points outliers detection and time series outliers detection. I am focusing on the time series related anomaly detection problems by using multiple distance measures based approaches in recent years.

PUBLICATIONS AND PAPERS:

- Huaming Huang and Kishan Mehrotra and Chilukuri K. Mohan, "Rank-based outlier detection", Journal of Statistical Computation and Simulation, vol82, 2011, 1-14
- Huaming Huang and Kishan Mehrotra and Chilukuri K. Mohan, "Algorithms for Detecting Outliers via Clustering and Ranks", IEA/AIE'12, The 25th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Dalian, China, 2012, 20-29.
- Huaming Huang and Kishan Mehrotra and Chilukuri K. Mohan, "An Online Anomalous Time Series Detection Algorithm for Univariate Data Streams", IEA/AIE'13, The 26th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Amsterdam, Netherlands, 2013.
- Huaming Huang and Kishan Mehrotra and Chilukuri K. Mohan, "Detection of Anomalous Time Series Based on Multiple Distance Measures", 28th International Conference on Computers and Their Applications (CATA-2013), Honolulu, Hawaii, USA, 2013

Appendix A

Experiments for Rank Based Detection Algorithm

Table A.1: Comparison of LOF, COF, INFLO and RBDA for k = 4, 5, 6 and 7 respectively for synthetic dataset 1. The highest values are marked as bold.

k=4	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	**1.000**	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	**1.000**
15	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	**1.000**	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	**1.000**
30	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	**1.000**	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	**1.000**

k=5	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	**1.000**
15	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	**1.000**
30	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	**1.000**

k=6	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	**6**	**0.60**	**1.00**	0.913	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	**1.000**
15	**6**	**0.40**	**1.00**	0.913	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	**1.000**
30	**6**	**0.20**	**1.00**	0.913	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	**1.000**

k=7	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	**6**	**0.60**	**1.00**	0.913	**6**	**0.60**	**1.00**	0.913	**6**	**0.60**	**1.00**	0.955	**6**	**0.60**	**1.00**	**1.000**
15	**6**	**0.40**	**1.00**	0.913	**6**	**0.40**	**1.00**	0.913	**6**	**0.40**	**1.00**	0.955	**6**	**0.40**	**1.00**	**1.000**
30	**6**	**0.20**	**1.00**	0.913	**6**	**0.20**	**1.00**	0.913	**6**	**0.20**	**1.00**	0.955	**6**	**0.20**	**1.00**	**1.000**

Table A.2: Comparison of LOF, COF, INFLO and RBDA for k = 25, 35 and 50 respectively for synthetic dataset 2. The highest values are marked as bold.

k=25	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	4	0.80	0.67	**1.000**	3	0.60	0.50	0.857	3	0.60	0.50	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	5	0.50	0.83	0.882	3	0.30	0.50	0.857	4	0.40	0.67	0.714	**6**	**0.60**	**1.00**	**1.000**
15	**6**	**0.40**	**1.00**	0.656	5	0.33	0.83	0.484	5	0.33	0.83	0.600	**6**	**0.40**	**1.00**	**1.000**
20	**6**	**0.30**	**1.00**	0.656	5	0.25	0.83	0.484	5	0.25	0.83	0.600	**6**	**0.30**	**1.00**	**1.000**
30	**6**	**0.20**	**1.00**	0.656	**6**	**0.20**	**1.00**	0.375	**6**	**0.20**	**1.00**	0.438	**6**	**0.20**	**1.00**	**1.000**

k=35	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	1	0.20	0.17	**1.000**	0	0.00	0.00	0.000	1	0.20	0.17	**1.000**	**5**	**1.00**	**0.83**	**1.000**
10	3	0.30	0.50	0.375	2	0.20	0.33	0.158	3	0.30	0.50	0.375	**5**	**0.50**	**0.83**	**1.000**
15	5	0.33	0.83	0.357	5	0.33	0.83	0.259	5	0.33	0.83	0.375	**6**	**0.40**	**1.00**	**0.808**
20	**6**	**0.30**	**1.00**	0.362	5	0.25	0.83	0.259	5	0.25	0.83	0.375	**6**	**0.30**	**1.00**	**0.808**
30	**6**	**0.20**	**1.00**	0.362	**6**	**0.20**	**1.00**	0.256	**6**	**0.20**	**1.00**	0.344	**6**	**0.20**	**1.00**	**0.808**

k=50	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	1	0.20	0.17	0.500	0	0.00	0.00	0.000	1	0.20	0.17	0.500	**5**	**1.00**	**0.83**	**1.000**
10	2	0.20	0.33	0.250	1	0.10	0.17	0.100	2	0.20	0.33	0.250	**5**	**0.50**	**0.83**	**1.000**
15	4	0.27	0.67	0.278	1	0.07	0.17	0.100	**5**	**0.33**	**0.83**	0.300	**5**	**0.33**	**0.83**	**1.000**
20	**6**	**0.30**	**1.00**	0.300	2	0.10	0.33	0.100	5	0.25	0.83	0.300	5	0.25	0.83	**1.000**
30	**6**	**0.20**	**1.00**	0.300	**6**	**0.20**	**1.00**	0.175	**6**	**0.20**	**1.00**	0.292	**6**	**0.20**	**1.00**	**0.583**

Table A.3: Comparison of LOF, COF, INFLO and RBDA for k = 11, 15, 20 and 23 respectively for the Ionosphere dataset. The highest values are marked as bold.

k=11	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**
15	6	0.40	0.60	**1.000**	6	0.40	0.60	0.778	6	0.40	0.60	**1.000**	**8**	**0.53**	**0.80**	0.818
30	7	0.23	0.70	0.667	7	0.23	0.70	0.560	7	0.23	0.70	0.651	**9**	**0.30**	**0.90**	**0.703**
60	8	0.13	0.80	0.409	8	0.13	0.80	0.409	8	0.13	0.80	0.419	**9**	**0.15**	**0.90**	**0.703**
85	9	0.11	0.90	0.294	9	0.11	0.90	0.290	9	0.11	0.90	0.300	**10**	**0.12**	**1.00**	**0.372**

k=15	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**
15	6	0.40	0.60	**1.000**	6	0.40	0.60	0.913	6	0.40	0.60	**1.000**	**8**	**0.53**	**0.80**	0.818
30	7	0.23	0.70	**0.757**	8	0.27	0.80	0.522	7	0.23	0.70	0.700	**9**	**0.30**	**0.90**	0.714
60	**9**	**0.15**	**0.90**	0.354	**9**	**0.15**	**0.90**	0.372	**9**	**0.15**	**0.90**	0.352	**9**	**0.15**	**0.90**	**0.714**
85	9	0.11	0.90	0.354	9	0.11	0.90	0.372	9	0.11	0.90	0.352	**10**	**0.12**	**1.00**	**0.377**

k=20	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**
15	7	0.47	0.70	**0.848**	7	0.47	0.70	0.737	7	0.47	0.70	0.800	**8**	**0.53**	**0.80**	0.837
30	7	0.23	0.70	**0.848**	7	0.23	0.70	0.737	7	0.23	0.70	0.800	**9**	**0.30**	**0.90**	0.738
60	**9**	**0.15**	**0.90**	0.417	**9**	**0.15**	**0.90**	0.413	**9**	**0.15**	**0.90**	0.405	**9**	**0.15**	**0.90**	**0.738**
85	9	0.11	0.90	**0.417**	9	0.11	0.90	0.413	9	0.11	0.90	0.405	**10**	**0.12**	**1.00**	0.387

k=25	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**	**5**	**1.00**	**0.50**	**1.000**
15	7	0.47	0.70	**0.875**	7	0.47	0.70	0.848	7	0.47	0.70	0.848	**8**	**0.53**	**0.80**	0.837
30	7	0.23	0.70	**0.875**	8	0.27	0.80	0.621	7	0.23	0.70	0.848	**9**	**0.30**	**0.90**	0.738
60	**9**	**0.15**	**0.90**	0.441	**9**	**0.15**	**0.90**	0.484	**9**	**0.15**	**0.90**	0.417	**9**	**0.15**	**0.90**	**0.738**
85	9	0.11	0.90	0.441	9	0.11	0.90	**0.484**	9	0.11	0.90	0.417	**10**	**0.12**	**1.00**	0.393

Table A.4: Comparison of LOF, COF, INFLO and RBDA for k = 11, 15, 19, and 22 respectively for the Wisconsin Breast Cancer data. The highest values are marked as bold.

$k = 11$	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
15	5	0.33	0.50	0.469	6	0.40	0.60	**0.656**	5	0.33	0.50	0.417	**7**	**0.47**	**0.70**	0.519
20	6	0.30	0.60	0.420	7	0.35	0.70	**0.560**	6	0.30	0.60	0.404	**9**	**0.45**	**0.90**	0.517
30	9	0.30	0.90	0.372	**10**	**0.33**	**1.00**	0.444	8	0.27	0.80	0.375	9	0.30	0.90	**0.517**
40	9	0.23	0.90	0.372	**10**	**0.25**	**1.00**	0.444	9	0.23	0.90	0.349	9	0.23	0.90	**0.517**
50	**10**	**0.20**	**1.00**	0.324	**10**	**0.20**	**1.00**	**0.444**	9	0.18	0.90	0.349	**10**	**0.20**	**1.00**	0.430

$k = 15$	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
15	6	0.40	0.60	0.525	**7**	**0.47**	**0.70**	**0.848**	6	0.40	0.60	0.538	**7**	**0.47**	**0.70**	0.596
20	7	0.35	0.70	0.467	**8**	**0.40**	**0.80**	**0.692**	6	0.30	0.60	0.538	**8**	**0.40**	**0.80**	0.554
30	**9**	**0.30**	**0.90**	0.395	8	0.27	0.80	0.692	8	0.27	0.80	0.439	**9**	**0.30**	**0.90**	**0.495**
40	9	0.23	0.90	0.395	**10**	**0.25**	**1.00**	0.474	9	0.23	0.90	0.381	9	0.23	0.90	**0.495**
50	**10**	**0.20**	**1.00**	0.346	**10**	**0.20**	**1.00**	**0.474**	9	0.18	0.90	0.381	**10**	**0.20**	**1.00**	0.407

$k = 19$	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
15	6	0.40	0.60	0.568	**7**	**0.47**	**0.70**	**0.848**	6	0.40	0.60	0.538	**7**	**0.47**	**0.70**	0.622
20	**8**	**0.40**	**0.80**	0.480	7	0.35	0.70	**0.848**	7	0.35	0.70	0.483	**8**	**0.40**	**0.80**	0.590
30	**9**	**0.30**	**0.90**	0.441	**9**	**0.30**	**0.90**	**0.536**	8	0.27	0.80	0.450	**9**	**0.30**	**0.90**	0.511
40	9	0.23	0.90	0.441	**10**	**0.25**	**1.00**	0.474	9	0.23	0.90	0.391	9	0.23	0.90	**0.511**
50	**10**	**0.20**	**1.00**	0.372	**10**	**0.20**	**1.00**	**0.474**	9	0.18	0.90	0.391	**10**	**0.20**	**1.00**	0.410

$k = 22$	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
15	**7**	**0.47**	**0.70**	0.549	**7**	**0.47**	**0.70**	**0.848**	**7**	**0.47**	**0.70**	0.538	**7**	**0.47**	**0.70**	0.636
20	7	0.35	0.70	0.549	7	0.35	0.70	**0.848**	7	0.35	0.70	0.538	**8**	**0.40**	**0.80**	0.600
30	**9**	**0.30**	**0.90**	0.446	**9**	**0.30**	**0.90**	0.500	8	0.27	0.80	0.486	**9**	**0.30**	**0.90**	**0.517**
40	9	0.23	0.90	0.446	**10**	**0.25**	**1.00**	0.451	9	0.23	0.90	0.417	9	0.23	0.90	**0.517**
50	**10**	**0.20**	**1.00**	0.374	**10**	**0.20**	**1.00**	**0.451**	9	0.18	0.90	0.417	**10**	**0.20**	**1.00**	0.414

Table A.5: Comparison of LOF, COF, INFLO and RBDA for k = 5, 7 and 10 respectively for the Iris dataset. The highest values are marked as bold.

m	LOF				COF				INFLO				RBDA			
	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**1**	**0.20**	**0.20**	**0.200**	0	0.00	0.00	0.000	**1**	**0.20**	**0.20**	**0.200**	**1**	**0.20**	**0.20**	**0.200**
10	**4**	**0.40**	**0.80**	0.313	0	0.00	0.00	0.000	2	0.20	0.40	0.214	**4**	**0.40**	**0.80**	**0.370**
15	**5**	**0.33**	**1.00**	0.341	0	0.00	0.00	0.000	2	0.13	0.40	0.214	**5**	**0.33**	**1.00**	**0.385**
20	**5**	**0.25**	**1.00**	0.341	0	0.00	0.00	0.000	2	0.10	0.40	0.214	**5**	**0.25**	**1.00**	**0.385**

m	LOF				COF				INFLO				RBDA			
	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**1.00**	**1.000**	0	0.00	0.00	0.000	3	0.60	0.60	0.667	3	0.60	0.60	0.750
10	**5**	**0.50**	**1.00**	**1.000**	0	0.00	0.00	0.000	4	0.40	0.80	0.625	**5**	**0.50**	**1.00**	0.714
15	**5**	**0.33**	**1.00**	**1.000**	2	0.13	0.40	0.103	**5**	**0.33**	**1.00**	0.556	**5**	**0.33**	**1.00**	0.714
20	**5**	**0.25**	**1.00**	**1.000**	3	0.15	0.60	0.130	**5**	**0.25**	**1.00**	0.556	**5**	**0.25**	**1.00**	0.714

m	LOF				COF				INFLO				RBDA			
	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**5**	**1.00**	**1.00**	**1.000**	0	0.00	0.00	0.000	**5**	**1.00**	**1.00**	**1.000**	**5**	**1.00**	**1.00**	**1.000**
10	**5**	**0.50**	**1.00**	**1.000**	2	0.20	0.40	0.176	**5**	**0.50**	**1.00**	**1.000**	**5**	**0.50**	**1.00**	**1.000**
15	**5**	**0.33**	**1.00**	**1.000**	**5**	**0.33**	**1.00**	0.278	**5**	**0.33**	**1.00**	**1.000**	**5**	**0.33**	**1.00**	**1.000**
20	**5**	**0.25**	**1.00**	**1.000**	**5**	**0.25**	**1.00**	0.278	**5**	**0.25**	**1.00**	**1.000**	**5**	**0.25**	**1.00**	**1.000**

Table A.6: Comparison of LOF, COF, INFLO and RBDA for k = 7 and 15, respectively, for the Iris data with planted anomalies. The highest values are marked as bold.

k=7	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**3**	**0.60**	**0.60**	**1.000**	2	0.40	0.40	1.000	**3**	**0.60**	**0.60**	**1.000**	**3**	**0.60**	**0.60**	**1.000**
10	**3**	**0.30**	**0.60**	**1.000**	**3**	**0.30**	**0.60**	0.667	**3**	**0.30**	**0.60**	**1.000**	**3**	**0.30**	**0.60**	**1.000**
15	**3**	**0.20**	**0.60**	**1.000**	**3**	**0.20**	**0.60**	0.667	**3**	**0.20**	**0.60**	**1.000**	**3**	**0.20**	**0.60**	**1.000**
20	**3**	**0.15**	**0.60**	**1.000**	**3**	**0.15**	**0.60**	0.667	**3**	**0.15**	**0.60**	**1.000**	**3**	**0.15**	**0.60**	**1.000**

k=15	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	**3**	**0.60**	**0.60**	0.857	**3**	**0.60**	**0.60**	0.857	**3**	**0.60**	**0.60**	**1.000**	**3**	**0.60**	**0.60**	**1.000**
10	**3**	**0.30**	**0.60**	0.857	**3**	**0.30**	**0.60**	0.857	**3**	**0.30**	**0.60**	**1.000**	**3**	**0.30**	**0.60**	**1.000**
15	**3**	**0.20**	**0.60**	0.857	**3**	**0.20**	**0.60**	0.857	**3**	**0.20**	**0.60**	**1.000**	**3**	**0.20**	**0.60**	**1.000**
20	**3**	**0.15**	**0.60**	0.857	**3**	**0.15**	**0.60**	0.857	**3**	**0.15**	**0.60**	**1.000**	**3**	**0.15**	**0.60**	**1.000**

Table A.7: Comparison of LOF, COF, INFLO and RBDA for k = 18, 25, and 35, respectively, for the Ionosphere data with planted anomalies. The highest values are marked as bold.

k=18	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
10	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000
20	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	**1**	**0.05**	**0.33**	**0.063**
30	1	0.03	0.33	0.037	0	0.00	0.00	0.000	**1**	**0.03**	**0.33**	0.038	**1**	**0.03**	**0.33**	**0.063**
40	1	0.03	0.33	0.037	1	0.03	0.33	0.026	2	0.05	0.67	0.046	**3**	**0.08**	**1.00**	**0.073**
50	2	0.04	0.67	0.042	2	0.04	0.67	0.034	2	0.04	0.67	0.046	**3**	**0.06**	**1.00**	**0.073**

k=25	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
10	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000
20	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	**1**	**0.05**	**0.33**	**0.077**
30	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	**1**	**0.03**	**0.33**	**0.077**
40	1	0.03	0.33	0.029	0	0.00	0.00	0.000	1	0.03	0.33	0.029	**3**	**0.08**	**1.00**	**0.075**
50	1	0.02	0.33	0.029	1	0.02	0.33	0.022	1	0.02	0.33	0.029	**3**	**0.06**	**1.00**	**0.075**

k=35	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
10	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000
20	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	**1**	**0.05**	**0.33**	**0.083**
30	0	0.00	0.00	0.000	0	0.00	0.00	0.000	0	0.00	0.00	0.000	**1**	**0.03**	**0.33**	**0.083**
40	1	0.03	0.33	0.031	1	0.03	0.33	0.029	1	0.03	0.33	0.029	**3**	**0.08**	**1.00**	**0.078**
50	1	0.02	0.33	0.031	1	0.02	0.33	0.029	1	0.02	0.33	0.029	**3**	**0.06**	**1.00**	**0.078**

Table A.8: Comparison of LOF, COF, INFLO and RBDA for k = 22, 35, and 45, respectively, for the Wisconsin Breast data with planted anomalies. The highest values are marked as bold.

k=22	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	1	0.20	0.50	**1.000**	1	0.20	0.50	0.500	1	0.20	0.50	**1.000**	**2**	**0.40**	**1.00**	**1.000**
10	1	0.10	0.50	**1.000**	**2**	**0.20**	**1.00**	0.273	1	0.10	0.50	**1.000**	**2**	**0.20**	**1.00**	**1.000**
20	1	0.05	0.50	**1.000**	**2**	**0.10**	**1.00**	0.273	1	0.05	0.50	**1.000**	**2**	**0.10**	**1.00**	**1.000**
30	1	0.03	0.50	**1.000**	**2**	**0.07**	**1.00**	0.273	**2**	**0.07**	**1.00**	0.130	**2**	**0.07**	**1.00**	**1.000**
40	**2**	**0.05**	**1.00**	0.077	**2**	**0.05**	**1.00**	0.273	**2**	**0.05**	**1.00**	0.130	**2**	**0.05**	**1.00**	**1.000**

k=35	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	1	0.20	0.50	**1.000**	**2**	**0.40**	**1.00**	0.429	1	0.20	0.50	**1.000**	**2**	**0.40**	**1.00**	**1.000**
10	1	0.10	0.50	**1.000**	**2**	**0.20**	**1.00**	0.429	**2**	**0.20**	**1.00**	0.300	**2**	**0.20**	**1.00**	**1.000**
15	**2**	**0.13**	**1.00**	0.231	**2**	**0.13**	**1.00**	0.429	**2**	**0.13**	**1.00**	0.300	**2**	**0.13**	**1.00**	**1.000**

k=45	LOF				COF				INFLO				RBDA			
m	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP	m_t	Pr	Re	RP
5	1	0.20	0.50	**1.000**	**2**	**0.40**	**1.00**	0.429	1	0.20	0.50	**1.000**	**2**	**0.40**	**1.00**	**1.000**
10	**2**	**0.20**	**1.00**	0.273	**2**	**0.20**	**1.00**	0.429	**2**	**0.20**	**1.00**	0.333	**2**	**0.20**	**1.00**	**1.000**
15	**2**	**0.13**	**1.00**	0.273	**2**	**0.13**	**1.00**	0.429	**2**	**0.13**	**1.00**	0.333	**2**	**0.13**	**1.00**	**1.000**

Appendix B

Experiments for Algorithms Based on Clustering and Weighted Ranks

Table B.1: Performance of all algorithms for synthetic dataset 2.
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=25	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
10	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
15	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
20	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
30	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**

k=25	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	**6**	**1**	**1**	3	0.5	0.857	4	0.667	1	3	0.5	0.857	3	0.5	1
10	**6**	**1**	**1**	4	0.667	0.667	5	0.833	0.882	3	0.5	0.857	4	0.667	0.714
15	**6**	**1**	**1**	5	0.833	0.577	**6**	**1**	0.656	5	0.833	0.484	5	0.833	0.6
20	**6**	**1**	**1**	**6**	**1**	0.5	**6**	**1**	0.656	5	0.833	0.484	5	0.833	0.6
30	**6**	**1**	**1**	**6**	**1**	0.5	**6**	**1**	0.656	**6**	**1**	0.375	**6**	**1**	0.438

k=35	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
10	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
15	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
20	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
30	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**

k=35	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	5	0.833	**1**	3	0.5	0.857	2	0.333	0.429	0	0	0	2	0.333	0.429
10	5	0.833	**1**	5	0.833	0.577	3	0.5	0.375	2	0.333	0.158	3	0.5	0.375
15	**6**	**1**	0.808	5	0.833	0.577	5	0.833	0.357	5	0.833	0.259	5	0.833	0.375
20	**6**	**1**	0.808	**6**	**1**	0.457	**6**	**1**	0.362	5	0.833	0.259	5	0.833	0.375
30	**6**	**1**	0.808	**6**	**1**	0.457	**6**	**1**	0.362	**6**	**1**	0.256	**6**	**1**	0.344

Table B.2: Performance of all algorithms for synthetic dataset 2 (Cont).
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=50	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
10	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
15	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
20	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**
30	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**	**6**	**1**	**1**

k=50	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
6	5	0.833	**1**	3	0.5	0.857	1	0.167	0.5	0	0	0	1	0.167	0.5
10	5	0.833	1	3	0.5	0.857	2	0.333	0.25	1	0.167	0.1	2	0.333	0.25
15	5	0.833	1	5	0.833	0.484	4	0.667	0.278	1	0.167	0.1	5	0.833	0.3
20	5	0.833	1	5	0.833	0.484	**6**	**1**	0.3	2	0.333	0.1	5	0.833	0.3
30	**6**	**1**	0.583	**6**	**1**	0.396	**6**	**1**	0.3	**6**	**1**	0.175	**6**	**1**	0.292

Table B.3: Performance of all algorithms for iris with rare class.
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=5	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	1	0.2	0.2	**3**	**0.6**	**0.75**	2	0.4	0.6	**3**	**0.6**	**0.857**	**3**	**0.6**	**0.75**
10	**5**	**1**	0.429	**5**	**1**	0.652	**5**	**1**	0.517	**5**	**1**	**0.75**	**5**	**1**	0.714
15	**5**	**1**	0.429	**5**	**1**	0.652	**5**	**1**	0.517	**5**	**1**	**0.75**	**5**	**1**	0.714
100	**5**	**1**	0.429	**5**	**1**	0.652	**5**	**1**	0.517	**5**	**1**	**0.75**	**5**	**1**	0.714

k=5	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	1	0.2	0.2	0	0	0	1	0.2	0.2	0	0	0	1	0.2	0.2
10	4	0.8	0.37	1	0.2	0.125	4	0.8	0.313	0	0	0	2	0.4	0.214
15	**5**	**1**	0.385	4	0.8	0.2	**5**	**1**	0.341	0	0	0	2	0.4	0.214
100	**5**	**1**	0.385	5	1	0.208	**5**	**1**	0.341	**5**	**1**	0.059	**5**	**1**	0.16

k=7	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	3	0.6	0.857	4	0.8	1	4	0.8	0.833	4	0.8	1	4	0.8	1
10	**5**	**1**	0.714	**5**	**1**	0.882	**5**	**1**	0.75	**5**	**1**	0.938	**5**	**1**	0.882
15	**5**	**1**	0.714	**5**	**1**	0.882	**5**	**1**	0.75	**5**	**1**	0.938	**5**	**1**	0.882
100	**5**	**1**	0.714	**5**	**1**	0.882	**5**	**1**	0.75	**5**	**1**	0.938	**5**	**1**	0.882

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	3	0.6	0.75	0	0	0	**5**	**1**	**1**	0	0	0	3	0.6	0.667
10	**5**	**1**	0.714	1	0.2	0.125	**5**	**1**	**1**	0	0	0	4	0.8	0.625
15	**5**	**1**	0.714	4	0.8	0.2	**5**	**1**	**1**	2	0.4	0.103	**5**	**1**	0.556
100	**5**	**1**	0.714	**5**	**1**	0.208	**5**	**1**	**1**	**5**	**1**	0.142	**5**	**1**	0.556

Table B.4: Performance of all algorithms for iris with rare class (Cont).
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=10	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**1**	**1**	4	0.8	**1**	4	0.8	**1**	**5**	**1**	**1**	4	0.8	**1**
10	**5**	**1**	**1**	**5**	**1**	0.938	**5**	**1**	0.882	**5**	**1**	**1**	**5**	**1**	0.882
15	**5**	**1**	**1**	**5**	**1**	0.938	**5**	**1**	0.882	**5**	**1**	**1**	**5**	**1**	0.882
100	**5**	**1**	**1**	**5**	**1**	0.938	**5**	**1**	0.882	**5**	**1**	**1**	**5**	**1**	0.882

k=10	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**1**	**1**	0	0	0	**5**	**1**	**1**	0	0	0	**5**	**1**	**1**
10	**5**	**1**	**1**	3	0.6	0.231	**5**	**1**	**1**	2	0.4	0.176	**5**	**1**	**1**
15	**5**	**1**	**1**	**5**	**1**	0.288	**5**	**1**	**1**	**5**	**1**	0.278	**5**	**1**	**1**
100	**5**	**1**	**1**	**5**	**1**	0.288	**5**	**1**	**1**	**5**	**1**	0.278	**5**	**1**	**1**

Table B.5: Performance of all algorithms for ionosphere dataset with rare class. The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=7	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.818	**8**	**0.8**	0.783	**8**	**0.8**	0.766	**8**	**0.8**	**0.837**	**8**	**0.8**	0.818
30	**9**	**0.9**	0.726	**9**	**0.9**	0.682	**9**	**0.9**	0.672	**9**	**0.9**	**0.738**	**9**	**0.9**	0.726
60	**9**	**0.9**	0.726	**9**	**0.9**	0.682	**9**	**0.9**	0.672	**9**	**0.9**	**0.738**	**9**	**0.9**	0.726
85	**10**	**1**	0.387	**10**	**1**	0.364	**10**	**1**	0.362	**10**	**1**	**0.396**	**10**	**1**	0.39

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	0	0	0	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.783	0	0	0	7	0.7	0.737	5	0.5	1	7	0.7	0.718
30	**9**	**0.9**	0.703	0	0	0	7	0.7	0.737	7	0.7	0.467	7	0.7	0.718
60	**9**	**0.9**	0.703	**9**	**0.9**	0.091	7	0.7	0.737	8	0.8	0.33	**9**	**0.9**	0.3
85	**10**	**1**	0.369	**10**	**1**	0.098	9	0.9	0.273	9	0.9	0.256	9	0.9	0.3

k=11	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.818	**8**	**0.8**	0.818	**8**	**0.8**	0.818	**8**	**0.8**	**0.837**	**8**	**0.8**	0.818
30	**9**	**0.9**	0.726	**9**	**0.9**	0.703	**9**	**0.9**	0.703	**9**	**0.9**	**0.738**	**9**	**0.9**	0.726
60	**9**	**0.9**	0.726	**9**	**0.9**	0.703	**9**	**0.9**	0.703	**9**	**0.9**	**0.738**	**9**	**0.9**	0.726
200	**10**	**1**	0.387	**10**	**1**	0.374	**10**	**1**	0.372	**10**	**1**	**0.399**	**10**	**1**	0.393

k=11	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	0	0	0	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.818	0	0	0	6	0.6	1	6	0.6	0.778	6	0.6	1
30	**9**	**0.9**	0.703	0	0	0	7	0.7	0.667	7	0.7	0.56	7	0.7	0.651
60	**9**	**0.9**	0.703	**9**	**0.9**	0.098	8	0.8	0.409	8	0.8	0.409	8	0.8	0.419
200	**10**	**1**	0.372	**10**	**1**	0.105	**10**	**1**	0.198	**10**	**1**	0.197	**10**	**1**	0.216

Table B.6: Performance of all algorithms for ionosphere dataset with rare class (Cont). The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=15	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.837	**8**	**0.8**	0.818	**8**	**0.8**	0.818	**8**	**0.8**	**0.857**	**8**	**0.8**	0.837
30	**9**	**0.9**	0.738	**9**	**0.9**	0.703	**9**	**0.9**	0.703	**9**	**0.9**	**0.75**	**9**	**0.9**	0.738
60	**9**	**0.9**	0.738	**9**	**0.9**	0.703	**9**	**0.9**	0.703	**9**	**0.9**	**0.75**	**9**	**0.9**	0.738
200	**10**	**1**	0.401	**10**	**1**	0.382	**10**	**1**	0.382	**10**	**1**	**0.423**	**10**	**1**	0.407

k=15	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	0	0	0	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.818	0	0	0	6	0.6	1	6	0.6	0.913	6	0.6	1
30	**9**	**0.9**	0.714	0	0	0	7	0.7	0.757	8	0.8	0.522	7	0.7	0.7
60	**9**	**0.9**	0.714	**9**	**0.9**	0.104	**9**	**0.9**	0.354	**9**	**0.9**	0.372	**9**	**0.9**	0.352
200	**10**	**1**	0.377	**10**	**1**	0.111	**10**	**1**	0.228	**10**	**1**	0.232	**10**	**1**	0.241

k=23	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	**0.857**	**8**	**0.8**	0.837	**8**	**0.8**	0.837	**8**	**0.8**	**0.857**	**8**	**0.8**	**0.857**
30	**9**	**0.9**	**0.75**	**9**	**0.9**	0.738	**9**	**0.9**	0.738	**9**	**0.9**	**0.75**	**9**	**0.9**	**0.75**
60	**9**	**0.9**	**0.75**	**9**	**0.9**	0.738	**9**	**0.9**	0.738	**9**	**0.9**	**0.75**	**9**	**0.9**	**0.75**
200	**10**	**1**	0.417	**10**	**1**	0.399	**10**	**1**	0.399	**10**	**1**	**0.43**	**10**	**1**	0.426

k=23	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
5	**5**	**0.5**	**1**	0	0	0	**5**	**0.5**	**1**	**5**	**0.5**	**1**	**5**	**0.5**	**1**
15	**8**	**0.8**	0.837	0	0	0	7	0.7	0.875	7	0.7	0.848	7	0.7	0.848
30	**9**	**0.9**	0.738	0	0	0	7	0.7	0.875	8	0.8	0.621	7	0.7	0.848
60	**9**	**0.9**	0.738	**9**	**0.9**	0.114	**9**	**0.9**	0.441	**9**	**0.9**	0.484	**9**	**0.9**	0.417
200	**10**	**1**	0.393	**10**	**1**	0.119	**10**	**1**	0.267	**10**	**1**	0.306	**10**	**1**	0.272

Table B.7: Comparison of all algorithms for the Wisconsin dataset with rare class. The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=7	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	8	0.8	**0.8**	7	0.7	**0.8**	7	0.7	**0.8**	8	0.8	0.735	8	0.8	**0.8**
25	**10**	**1**	**0.64**	**10**	**1**	0.611	**10**	**1**	0.598	9	0.9	0.643	**10**	**1**	0.618
40	**10**	**1**	**0.64**	**10**	**1**	0.611	**10**	**1**	0.598	**10**	**1**	0.573	**10**	**1**	0.618

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	**9**	**0.9**	0.714	**9**	**0.9**	0.662	5	0.5	0.306	6	0.6	0.42	5	0.5	0.326
25	**10**	**1**	**0.64**	9	0.9	0.662	7	0.7	0.318	**10**	**1**	0.417	7	0.7	0.318
40	**10**	**1**	**0.64**	**10**	**1**	0.545	**10**	**1**	0.288	**10**	**1**	0.417	**10**	**1**	0.314

k=11	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	8	0.8	**0.783**	8	0.8	0.655	8	0.8	0.667	8	0.8	0.706	8	0.8	0.735
25	**10**	**1**	0.604	**10**	**1**	0.579	10	1	0.591	9	0.9	0.592	**10**	**1**	0.573
40	**10**	**1**	0.604	**10**	**1**	0.579	**10**	**1**	0.591	**10**	**1**	0.534	**10**	**1**	0.573

k=11	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	7	0.7	0.651	**9**	**0.9**	0.763	6	0.6	0.5	9	0.9	0.703	6	0.6	0.538
25	**10**	**1**	0.573	9	0.9	**0.763**	9	0.9	0.409	**10**	**1**	0.655	8	0.8	0.45
40	**10**	**1**	0.573	**10**	**1**	0.598	**10**	**1**	0.385	**10**	**1**	**0.655**	**10**	**1**	0.39

k=22	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	8	0.8	0.75	8	0.8	0.735	8	0.8	0.72	8	0.8	0.72	8	0.8	0.75
25	**10**	**1**	0.579	**10**	**1**	0.585	9	0.9	0.643	9	0.9	0.616	10	1	0.573
40	**10**	**1**	0.579	**10**	**1**	0.585	**10**	**1**	0.573	**10**	**1**	0.55	**10**	**1**	0.573

k=22	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
15	8	0.8	0.667	**9**	**0.9**	**0.804**	8	0.8	0.581	8	0.8	0.878	7	0.7	0.596
25	9	0.9	0.634	9	0.9	**0.804**	9	0.9	0.549	9	0.9	0.692	9	0.9	0.529
40	**10**	**1**	0.567	**10**	**1**	**0.625**	**10**	**1**	0.505	**10**	**1**	0.585	**10**	**1**	0.466

Table B.8: Comparison of all algorithms for the iris dataset with planted outliers. The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=7	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**3**	**1**	**1**	**3**	**1**	**1**	**3**	**1**	**1**	**3**	**1**	**1**	**3**	**1**	**1**

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**3**	**1**	**1**	**3**	**1**	**1**	**3**	**1**	**1**	**3**	**1**	0.667	**3**	**1**	**1**

Table B.9: Comparison of all algorithms for the ionosphere dataset with planted outliers. The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=7	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	0.091	**1**	**0.333**	0.059	**1**	**0.333**	0.059	**1**	**0.333**	0.091	**1**	**0.333**	0.083
30	1	0.333	0.091	**2**	**0.667**	0.068	**2**	**0.667**	0.068	**2**	**0.667**	0.075	1	0.333	0.083
40	**3**	**1**	0.077	**3**	**1**	0.072	**3**	**1**	0.072	**3**	**1**	0.081	**3**	**1**	0.076

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	**1**	**0.333**	0.333	**1**	**0.333**	**1**	**1**	**0.333**	0.333
20	**1**	**0.333**	0.059	0	0	0	**1**	**0.333**	0.333	**1**	**0.333**	**1**	**1**	**0.333**	0.333
30	**2**	**0.667**	0.068	0	0	0	1	0.333	0.333	1	0.333	**1**	1	0.333	0.333
40	**3**	**1**	0.072	0	0	0	1	0.333	0.333	1	0.333	**1**	1	0.333	0.333

k=18	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	0.083	**1**	**0.333**	0.063	**1**	**0.333**	0.063	**1**	**0.333**	**0.091**	**1**	**0.333**	0.083
30	1	0.333	**0.083**	1	0.333	0.063	1	0.333	0.063	**2**	**0.667**	0.073	1	0.333	0.083
40	**3**	**1**	0.077	**3**	**1**	0.073	**3**	**1**	0.073	**3**	**1**	**0.08**	**3**	**1**	0.077

k=18	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	0.063	0	0	0	0	0	0	0	0	0	0	0	0
30	1	0.333	0.063	0	0	0	1	0.333	0.037	0	0	0	1	0.333	0.038
40	**3**	**1**	0.073	0	0	0	1	0.333	0.037	1	0.333	0.026	2	0.667	0.046

Table B.10: Comparison of all algorithms for the ionosphere dataset with planted outliers (Cont).
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=25	RADA			ODMR			ODMRS			ODMRW			OMDRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	**0.091**	**1**	**0.333**	0.077	**1**	**0.333**	0.077	**1**	**0.333**	**0.091**	**1**	**0.333**	0.083
30	1	0.333	**0.091**	1	0.333	0.077	1	0.333	0.077	**2**	**0.667**	0.073	1	0.333	0.083
40	**3**	**1**	0.078	**3**	**1**	0.075	**3**	**1**	0.075	**3**	**1**	**0.081**	**3**	**1**	0.077

k=25	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	0.077	0	0	0	0	0	0	0	0	0	0	0	0
30	1	0.333	0.077	0	0	0	0	0	0	0	0	0	0	0	0
40	**3**	**1**	0.075	0	0	0	1	0.333	0.029	0	0	0	1	0.333	0.029

k=35	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	**0.091**	**1**	**0.333**	0.083	**1**	**0.333**	0.083	**1**	**0.333**	**0.091**	**1**	**0.333**	**0.091**
30	**2**	**0.667**	0.073	1	0.333	**0.083**	1	0.333	**0.083**	**2**	**0.667**	0.075	**2**	**0.667**	0.073
40	**3**	**1**	0.08	**3**	**1**	0.078	**3**	**1**	0.078	**3**	**1**	**0.082**	**3**	**1**	0.08

k=35	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	**1**	**0.333**	0.083	0	0	0	0	0	0	0	0	0	0	0	0
30	1	0.333	**0.083**	0	0	0	0	0	0	0	0	0	0	0	0
40	**3**	**1**	0.078	0	0	0	1	0.333	0.031	1	0.333	0.029	1	0.333	0.029

Table B.11: Comparison of all algorithms for the Wisconsin dataset with planted outliers (Cont).
The highest values among all algorithms for same measurement for the same k and m are shown in bold.

k=7	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
20	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
30	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
40	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**

k=7	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**2**	**1**	**1**	**2**	**1**	0.188	0	0	0	1	0.5	0.1	1	0.5	0.1
20	**2**	**1**	**1**	**2**	**1**	0.188	1	0.5	0.05	**2**	**1**	0.136	**2**	**1**	0.1
30	**2**	**1**	**1**	**2**	**1**	0.188	1	0.5	0.05	**2**	**1**	0.136	**2**	**1**	0.1
40	**2**	**1**	**1**	**2**	**1**	0.188	**2**	**1**	0.053	**2**	**1**	0.136	**2**	**1**	0.1

k=22	RADA			ODMR			ODMRS			ODMRW			ODMRD		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
20	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
30	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**
40	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**	**2**	**1**	**1**

k=22	RBDA			DBCOD			LOF			COF			INFLO		
m	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP	m_t	Re	RP
10	**2**	**1**	**1**	**2**	**1**	0.75	1	0.5	**1**	**2**	**1**	0.273	1	0.5	**1**
20	**2**	**1**	**1**	**2**	**1**	0.75	1	0.5	**1**	**2**	**1**	0.273	1	0.5	**1**
30	**2**	**1**	**1**	**2**	**1**	0.75	1	0.5	**1**	**2**	**1**	0.273	**2**	**1**	0.13
40	**2**	**1**	**1**	**2**	**1**	0.75	**2**	**1**	0.077	**2**	**1**	0.273	**2**	**1**	0.13

APPENDIX C
EXPERIMENTS FOR RELATIONSHIP BETWEEN SIZE OF SUBSEQUENCE AND PERFORMANCE IN SAXFR

C.1 Data set: SYN0

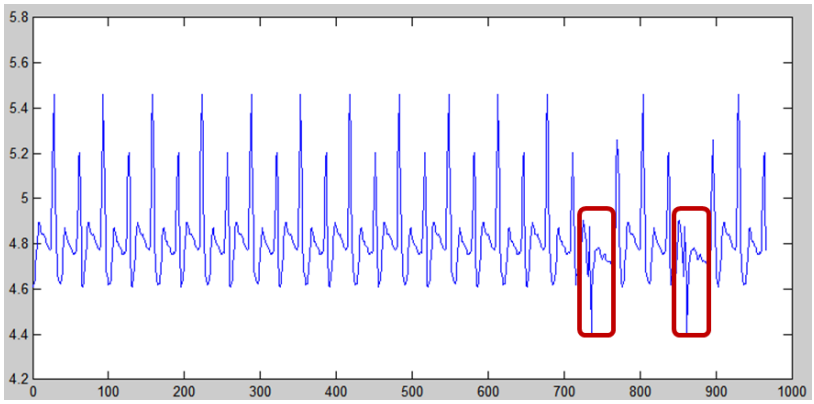

Fig. C.1: **Synthetic data set SYN0** - Abnormal subsequence is highlighted in red rectangle.

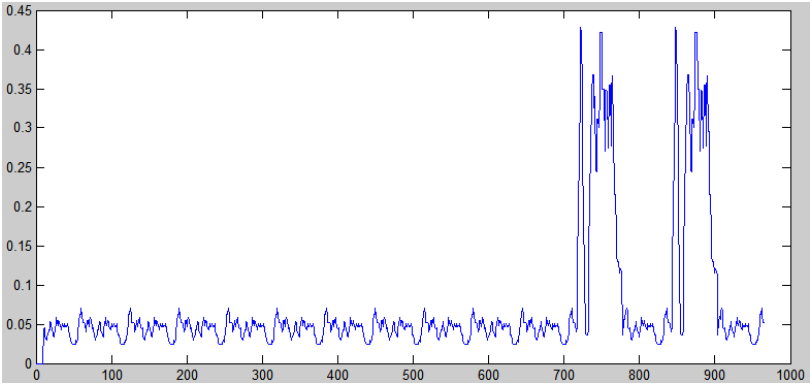

Fig. C.2: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 10. Y axis presents SAXFR values for each subsequence.

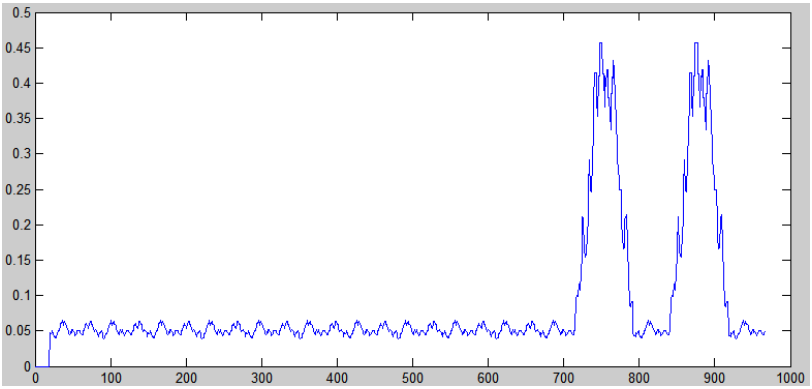

Fig. C.3: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 20. Y axis presents SAXFR values for each subsequence.

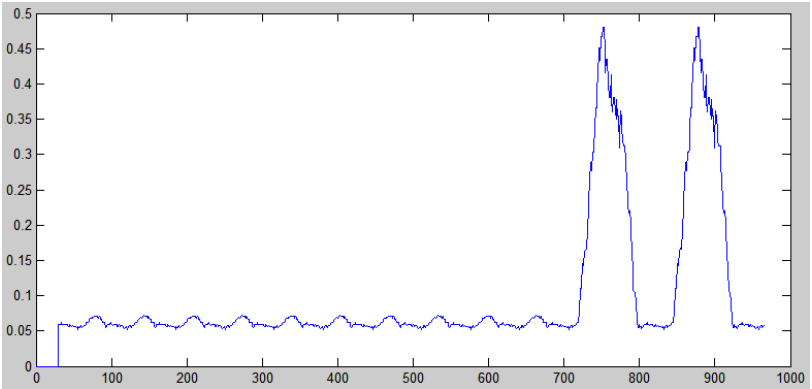

Fig. C.4: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 30. Y axis presents SAXFR values for each subsequence.

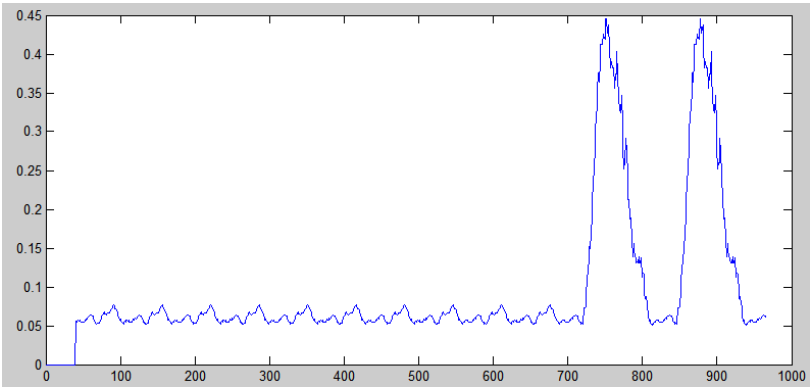

Fig. C.5: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 40. Y axis presents SAXFR values for each subsequence.

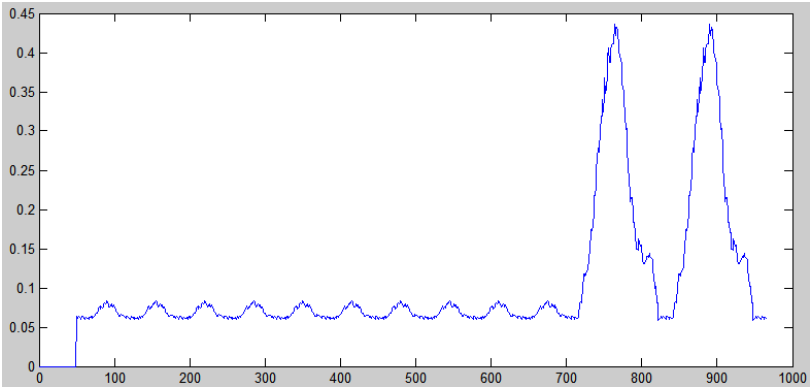

Fig. C.6: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 50. Y axis presents SAXFR values for each subsequence.

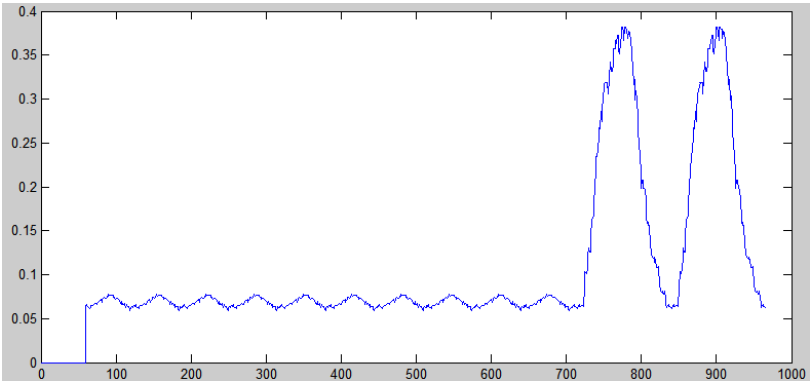

Fig. C.7: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 60. Y axis presents SAXFR values for each subsequence.

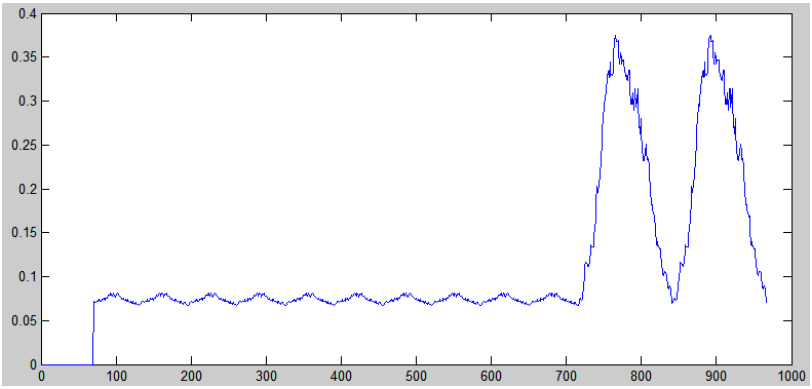

Fig. C.8: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 70. Y axis presents SAXFR values for each subsequence.

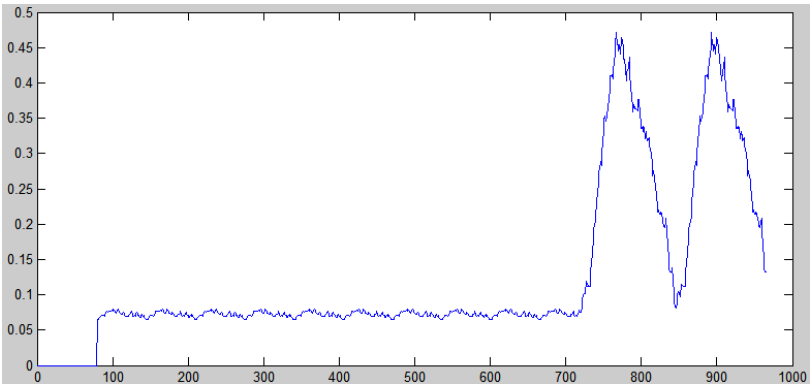

Fig. C.9: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 80. Y axis presents SAXFR values for each subsequence.

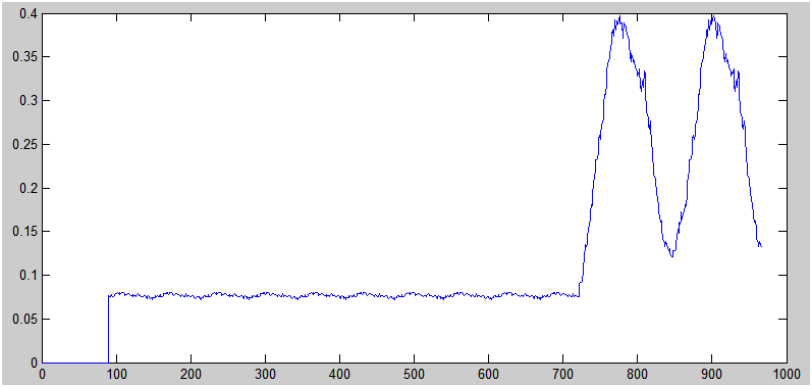

Fig. C.10: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 90. Y axis presents SAXFR values for each subsequence.

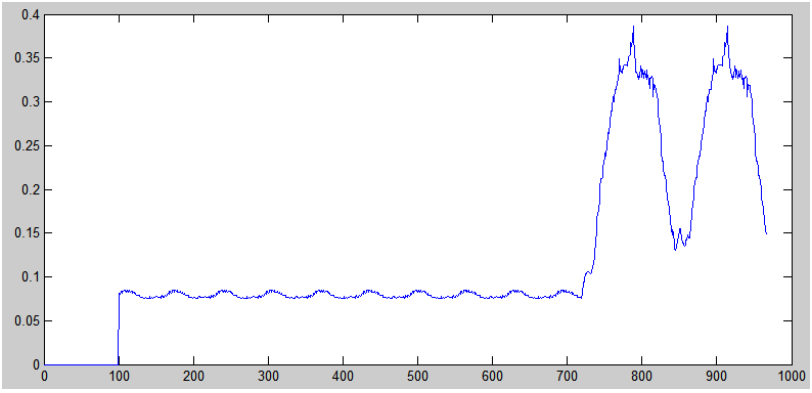

Fig. C.11: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 100. Y axis presents SAXFR values for each subsequence.

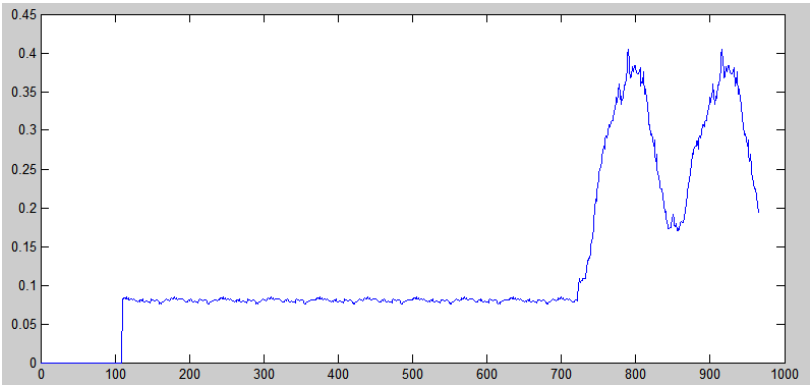

Fig. C.12: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 110. Y axis presents SAXFR values for each subsequence.

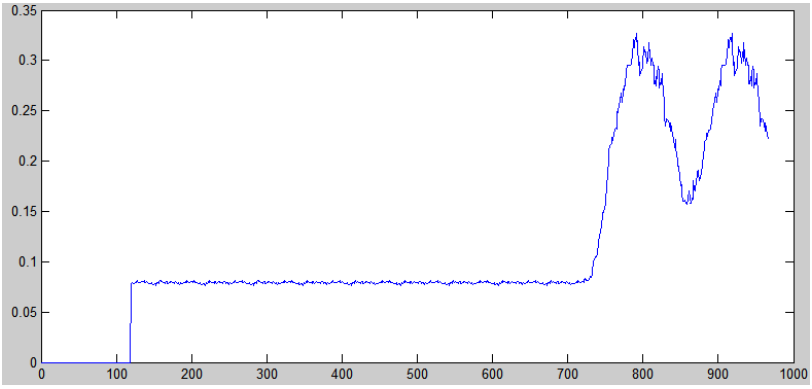

Fig. C.13: **SAXFR results for SYN0** - SAXFR results when size of subsequence is 120. Y axis presents SAXFR values for each subsequence.

C.2 Data set: ECG1

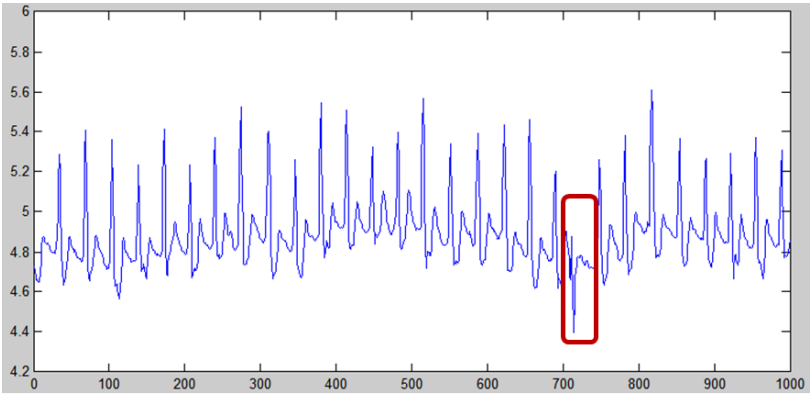

Fig. C.14: **Synthetic data set ECG1** - Abnormal subsequence is highlighted in red rectangle.

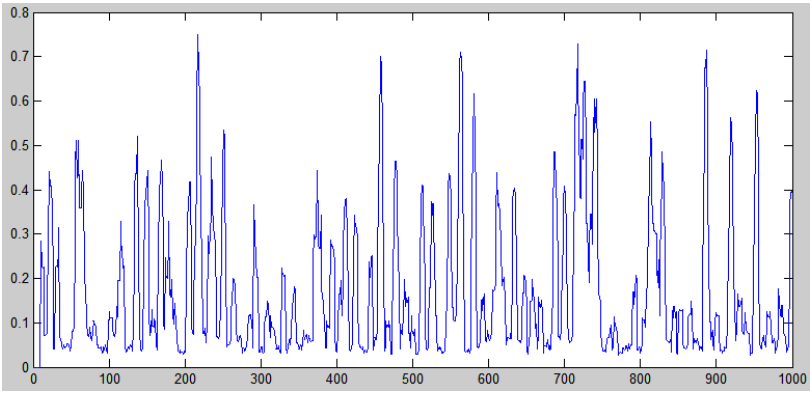

Fig. C.15: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 10. Y axis presents SAXFR values for each subsequence.

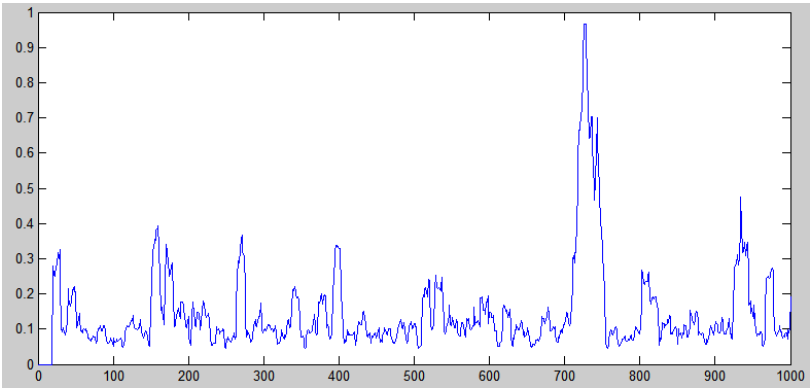

Fig. C.16: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 20. Y axis presents SAXFR values for each subsequence.

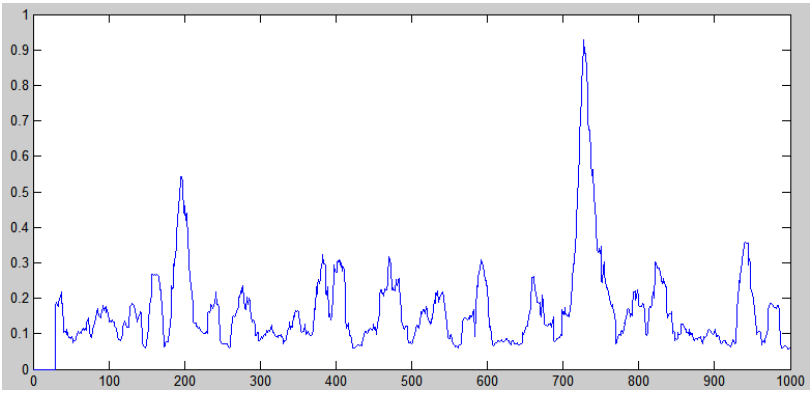

Fig. C.17: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 30. Y axis presents SAXFR values for each subsequence.

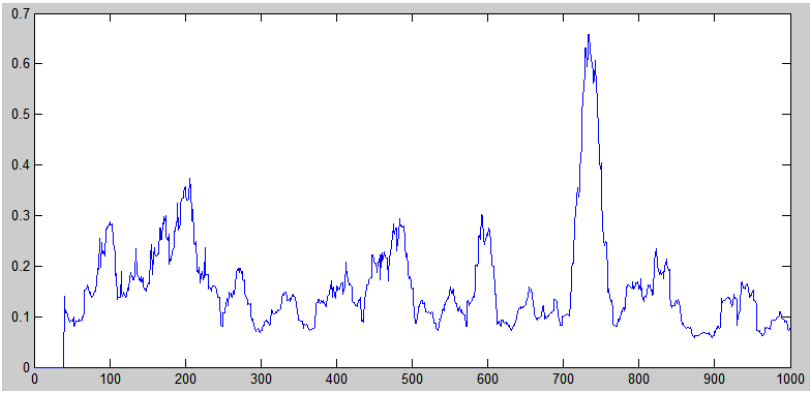

Fig. C.18: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 40. Y axis presents SAXFR values for each subsequence.

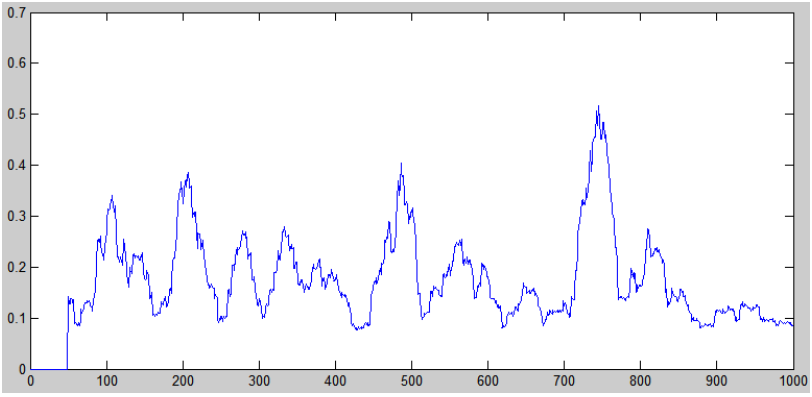

Fig. C.19: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 50. Y axis presents SAXFR values for each subsequence.

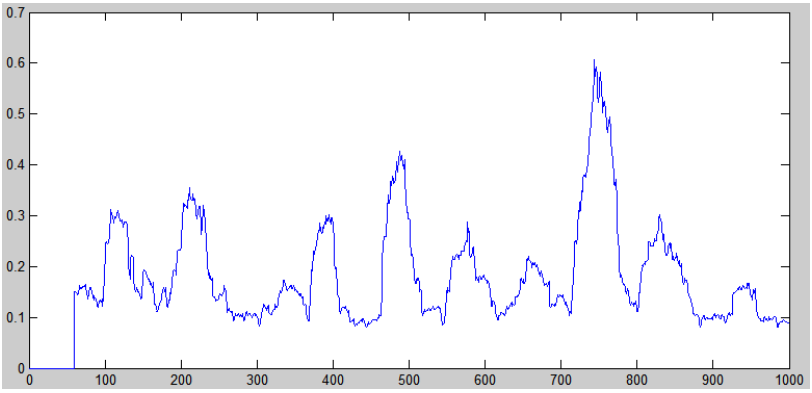

Fig. C.20: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 60. Y axis presents SAXFR values for each subsequence.

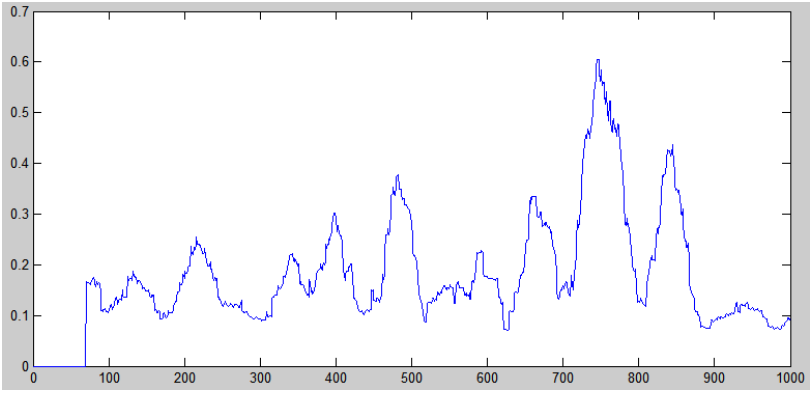

Fig. C.21: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 70. Y axis presents SAXFR values for each subsequence.

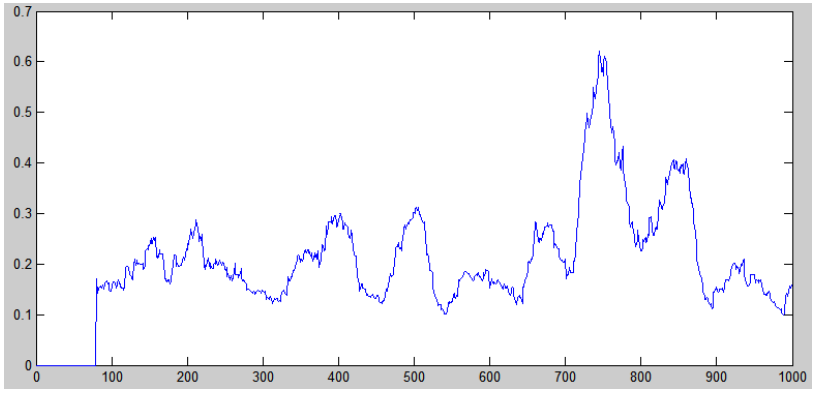

Fig. C.22: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 80. Y axis presents SAXFR values for each subsequence.

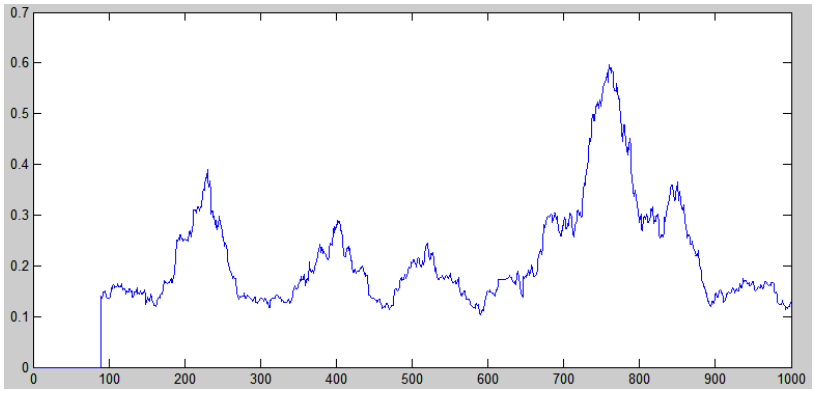

Fig. C.23: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 90. Y axis presents SAXFR values for each subsequence.

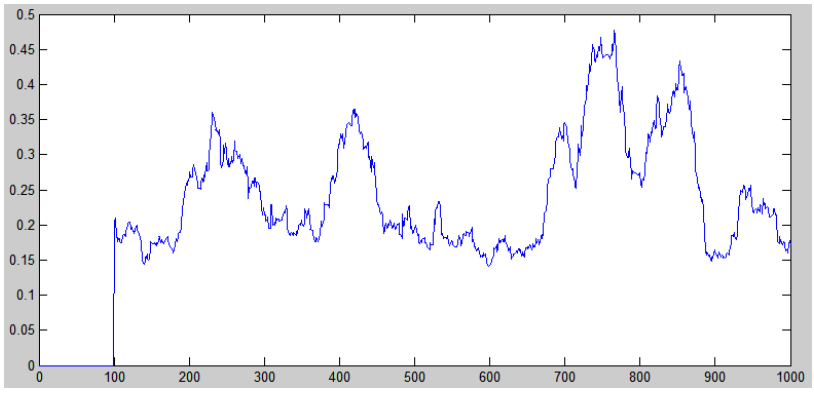

Fig. C.24: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 100. Y axis presents SAXFR values for each subsequence.

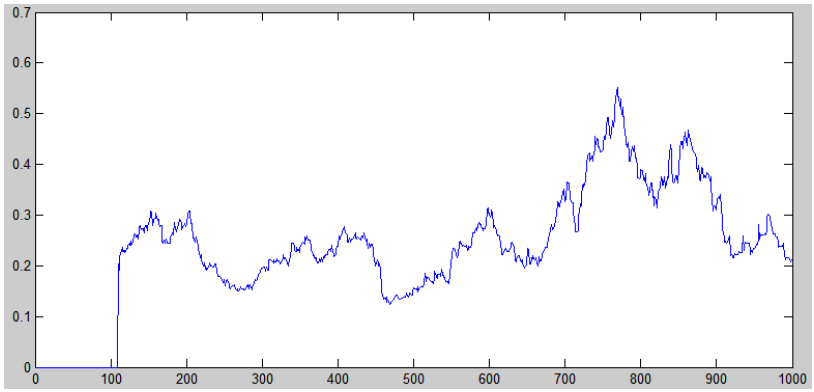

Fig. C.25: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 110. Y axis presents SAXFR values for each subsequence.

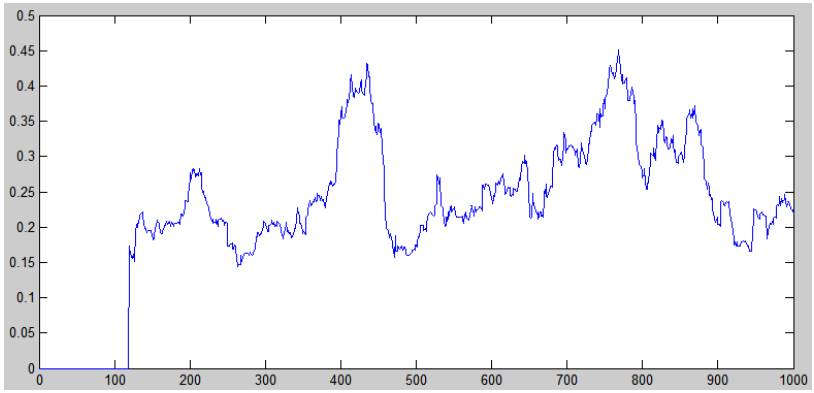

Fig. C.26: **SAXFR results for ECG1** - SAXFR results when size of subsequence is 120. Y axis presents SAXFR values for each subsequence.

C.3 Data set: ECG2

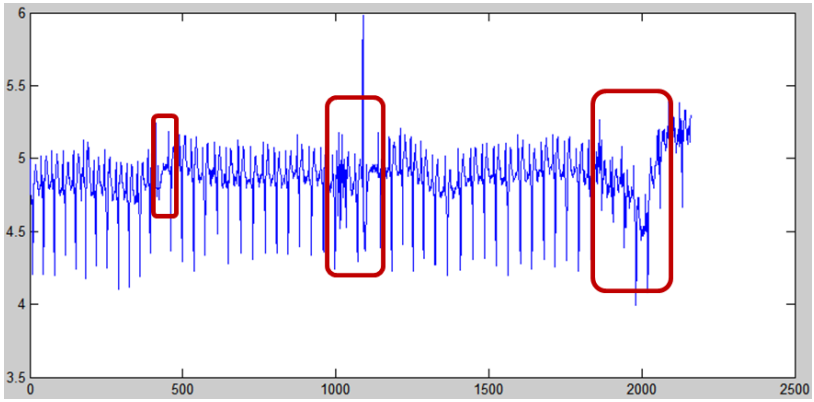

Fig. C.27: **Synthetic data set ECG2** - Abnormal subsequence is highlighted in red rectangle.

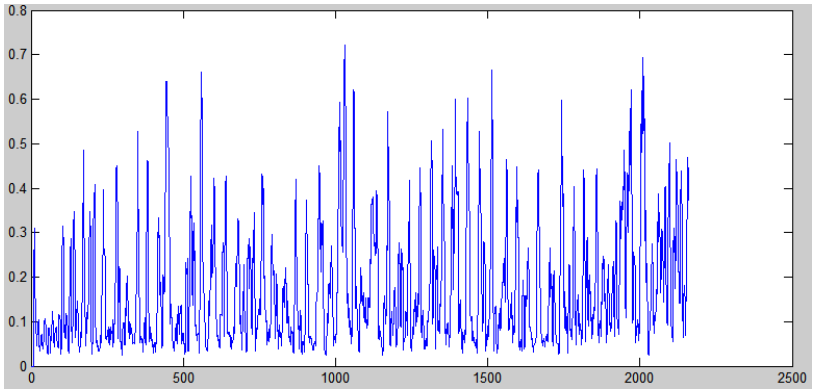

Fig. C.28: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 10. Y axis presents SAXFR values for each subsequence.

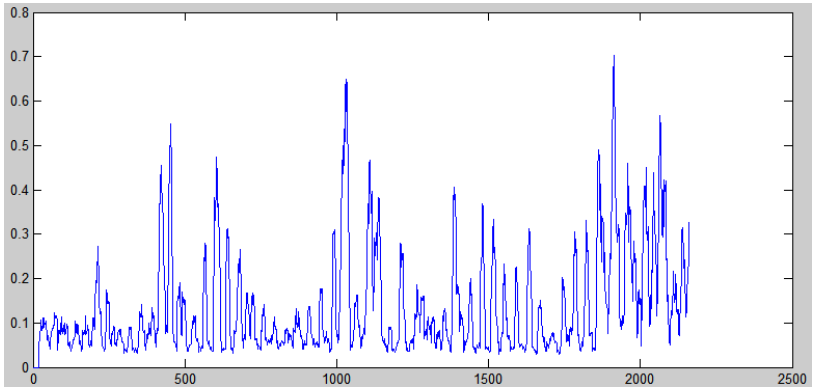

Fig. C.29: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 20. Y axis presents SAXFR values for each subsequence.

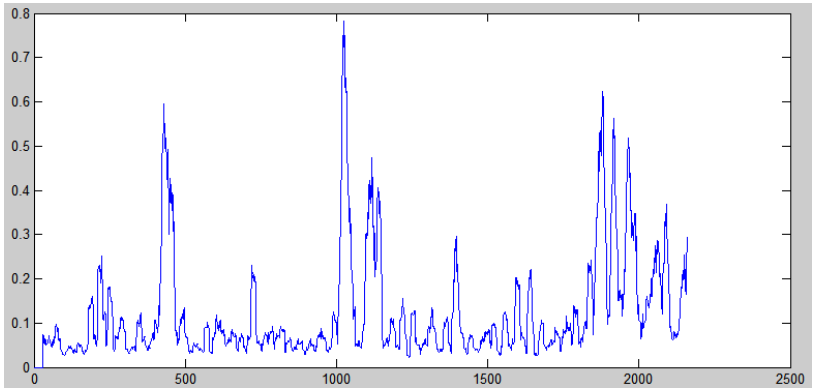

Fig. C.30: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 30. Y axis presents SAXFR values for each subsequence.

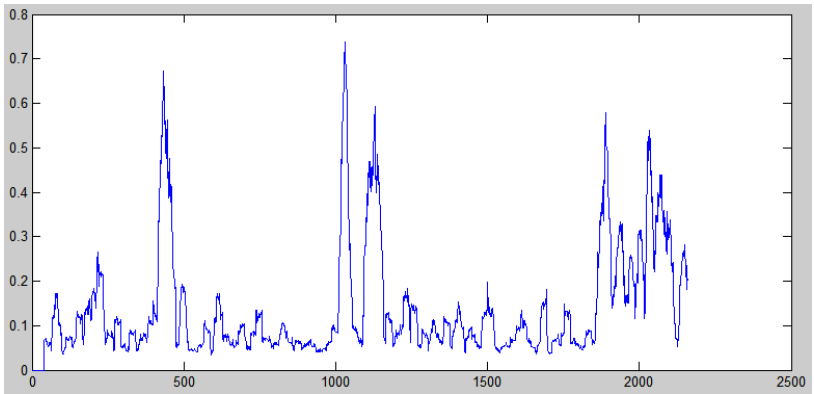

Fig. C.31: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 40. Y axis presents SAXFR values for each subsequence.

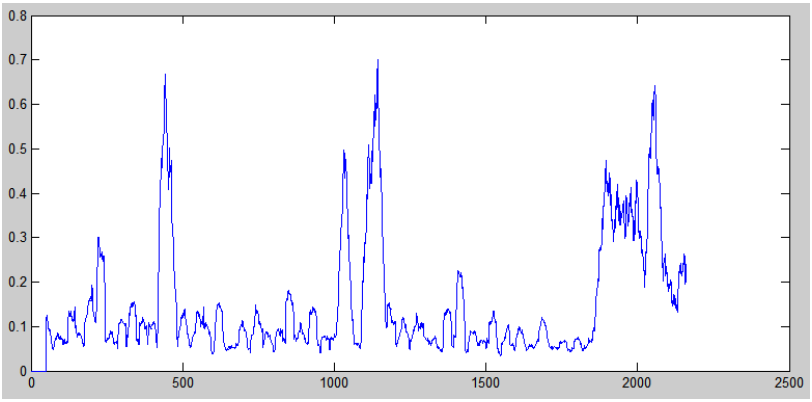

Fig. C.32: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 50. Y axis presents SAXFR values for each subsequence.

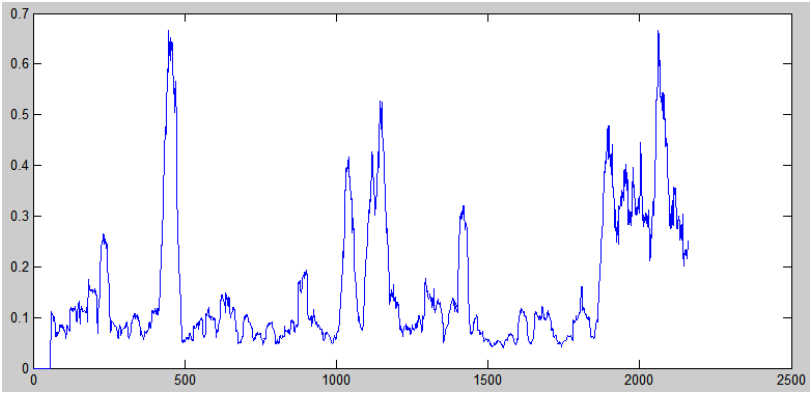

Fig. C.33: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 60. Y axis presents SAXFR values for each subsequence.

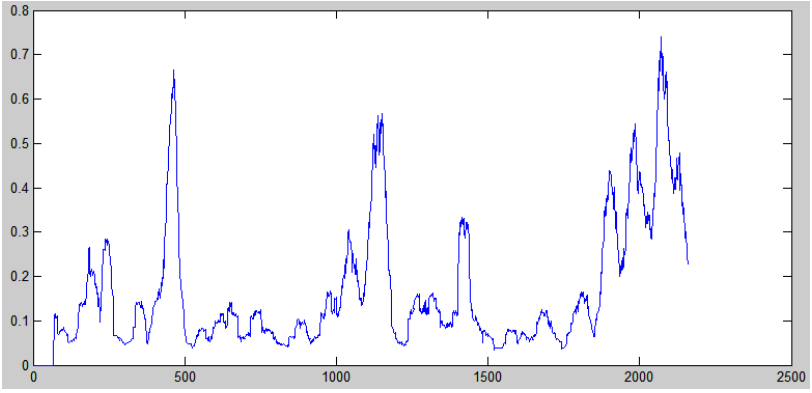

Fig. C.34: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 70. Y axis presents SAXFR values for each subsequence.

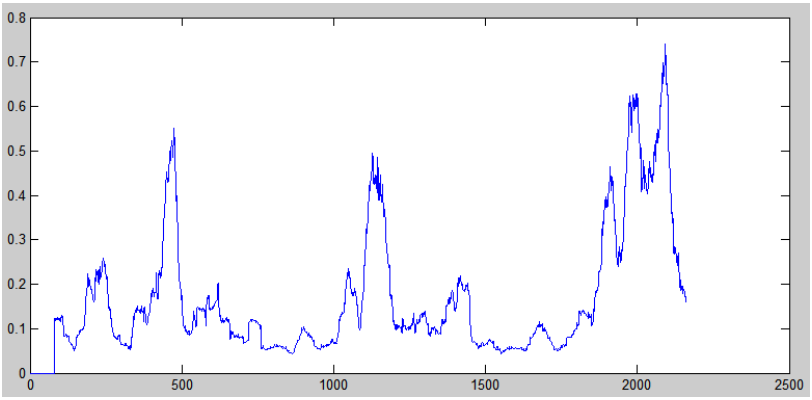

Fig. C.35: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 80. Y axis presents SAXFR values for each subsequence.

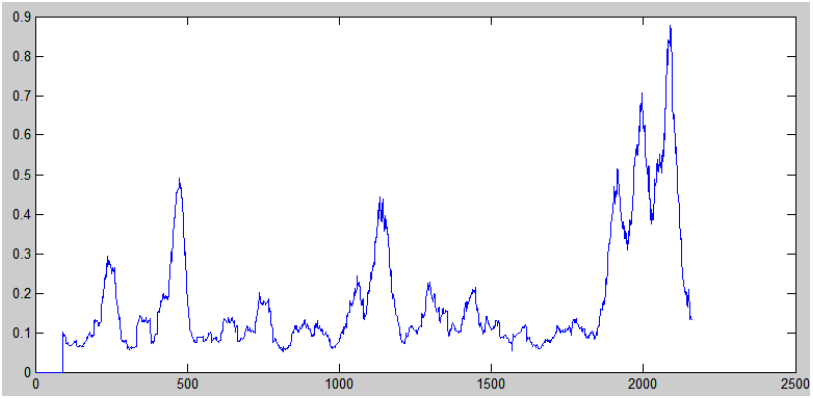

Fig. C.36: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 90. Y axis presents SAXFR values for each subsequence.

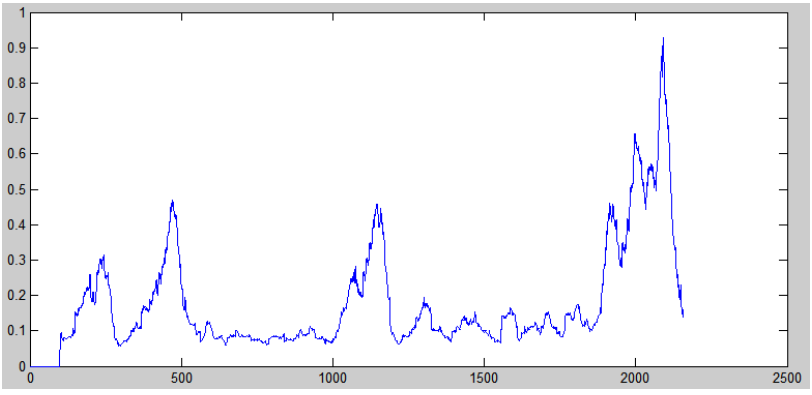

Fig. C.37: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 100. Y axis presents SAXFR values for each subsequence.

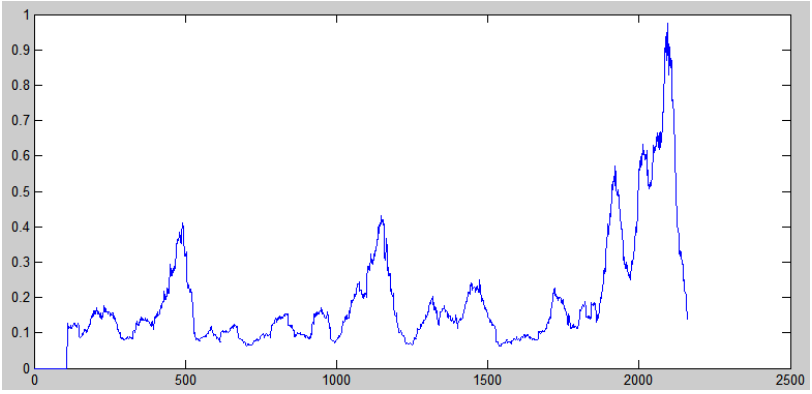

Fig. C.38: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 110. Y axis presents SAXFR values for each subsequence.

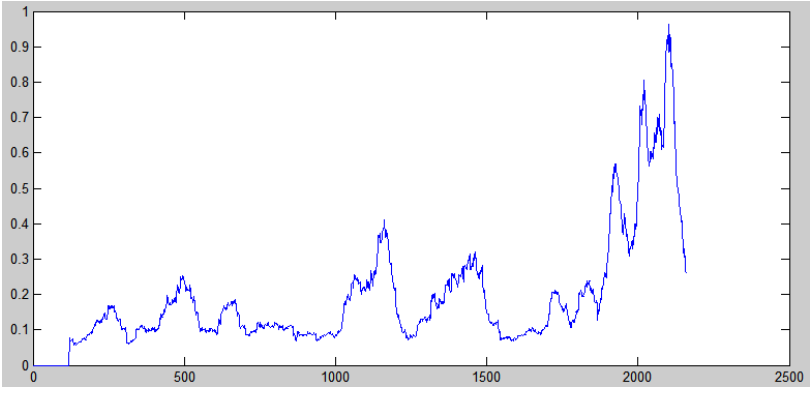

Fig. C.39: **SAXFR results for ECG2** - SAXFR results when size of subsequence is 120. Y axis presents SAXFR values for each subsequence.