Anomaly Detection for Symbolic Sequences and Time Series Data

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Varun Chandola

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor Of Philosophy

September, 2009

Acknowledgements

I am deeply indebted to my advisor, Prof. Vipin Kumar, whose constant guidance has enabled me to pursue and attain my research goals, eventually leading to this thesis. His support and enthusiasm for my research has been instrumental in making my journey, as a PhD student, possible. I have been greatly influenced by Prof. Kumar's work ethic and his enthusiasm towards scientific research. I have learnt the virtues of patience and perseverance while conducting research. He has taught me to focus on solutions that have an impact on human life. I have written this thesis based on his principle of developing practical solutions for real life problems.

My co-advisor, Dr. Arindam Banerjee, has not only guided me in my research at every step, but also helped me hone my technical writing skills; an invaluable learning that will help me throughout my research career. I thank him for his support and advice.

I extend my gratitude to my teachers at University of Minnesota, especially Prof. Jaideep Srivastava, Prof. Zhili Zhang, and Prof. Yongdae Kim, with whom I had the privilege of collaborating on several research projects. I would also like to thank my other collaborators at UMN, Dr. Michael Steinbac, Dr. Pang Ning Tan, Dr. Aleksandar Lazarevic, Levent Ertoz, and Gyorgy Simon. Special thanks to Shyam Boriah for all his help during this time and the highly productive research collaborations with him that form an important part of this thesis. I would also like to thank my thesis committee members for their insightful comments, which have helped me improve my thesis. I have had the good fortune of working at various institutions, where I evaluated the potential of my research in a real environment. I would like to thank Dr. Ashok Srivastava and Dr. Nikunj Oza, NASA Ames Research Center, Prof. Hillol Kargupta, Agnik Corp., Dr. Pavel Berkhin and Dr. Vijay Narayanan, Yahoo Inc., and Dr. Luc Mongeon, Medtronic

Inc. I am also appreciative of the Computer Science and AHPCRC administrative staff, especially Liz and Peggy, for their help in all day to day matters. During my stay at UMN, I have interacted with several bright students while writing papers and proposals. I thank Varun, Deepthi, Gaurav, Rohit, Gang, Gowtham, Tushar, and others. Special thanks to Devdutta, and other friends at UMN, who have helped me in different ways during the course of my PhD. Some of the research in this thesis has been funded by grants from various agencies, including NASA, NSF, DHS, AHPCRC, and ARDA; I am grateful for their support.

This thesis could not have been completed without the immense support of my parents, my brothers, and other family members. Finally, the most important ingredient for the completion of this thesis has been the love and support of Neeti, my wife and partner in my research endeavors, whose unending motivation and encouragement have been crucial in overcoming this daunting task.

Dedication

To my grandparents and parents.

ABSTRACT

This thesis deals with the problem of anomaly detection for sequence data. Anomaly detection has been a widely researched problem in several application domains such as system health management, intrusion detection, health-care, bio-informatics, fraud detection, and mechanical fault detection. Traditional anomaly detection techniques analyze each data instance (as a univariate or multivariate record) independently, and ignore the sequential aspect of the data. Often, anomalies in sequences can be detected only by analyzing data instances together as a sequence, and hence cannot detected by traditional anomaly detection techniques. The problem of anomaly detection for sequence data is a rich area of research because of two main reasons. First, sequences can be of different types, e.g., symbolic sequences, time series data, etc., and each type of sequence poses unique set of problems. Second, anomalies in sequences can be defined in multiple ways and hence there are different problem formulations. In this thesis we focus on solving one particular problem formulation called *semi-supervised anomaly detection*. We study the problem separately for symbolic sequences, univariate time series data, and multivariate time series data.

The state of art on anomaly detection for sequences is limited and fragmented across application domains. For symbolic sequences, several techniques have been proposed within specific domains, but it is not well-understood as to how a technique developed for one domain would perform in a completely different domain. For univariate time series data, limited techniques exist, and are only evaluated for specific domains, while for multivariate time series data, anomaly detection research is relatively untouched.

This thesis has two key goals. First goal is to develop novel anomaly detection techniques for different types of sequences which perform better than existing techniques across a variety of application domains. The second goal is to identify the best anomaly detection technique for a given application domain. By realizing the first goal, we develop a suite of anomaly detection techniques for a domain scientist to choose from, while the second goal will help the scientist to choose the technique best suited for the task.

To achieve the first goal, we develop several novel anomaly detection techniques for univariate symbolic sequences, univariate time series data, and multivariate time series data. We provide extensive experimental evaluation of the proposed techniques on data sets collected across diverse domains and generated from data generators, also developed as part of this thesis. We show how the proposed techniques can be used to detect anomalies which translate to critical events in domains such as aircraft safety, intrusion detection, and patient health management. The techniques proposed in this thesis are shown to outperform existing techniques on many data sets. The technique proposed for multivariate time series data is one of the very first anomaly detection technique that can detect complex anomalies in such data.

To achieve the second goal, we study the relationship between anomaly detection techniques and the nature of the data on which they are applied. A novel analysis framework, *Reference Based Analysis* (RBA), is proposed that can map a given data set (of any type) into a multivariate continuous space with respect to a reference data set. We apply the RBA framework to not only visualize and understand complex data types, such as multivariate categorical data and symbolic sequence data, but also to extract data driven features from symbolic sequences, which when used with traditional anomaly detection techniques are shown to consistently outperform the state of art anomaly detection techniques for these complex data types. Two novel techniques for symbolic sequences, WIN_{1D} and WIN_{2D} are proposed using the RBA framework which perform better than the best technique for each different data set.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Anomaly detection, loosely defined as the task of detecting abnormal patterns in data, is a significant research problem that has been widely explored in a variety of application domains [35, 84]. Anomalies in such domains typically correspond to highly critical events that are not observed in data corresponding to the normal behavior experienced in the given domain. In many domains, such as flight safety, intrusion detection, fraud detection, health-care, etc., data is collected in the form of sequences or time-series. For example, in the domain of aviation or flight safety, the data collected from flights is in the form of sequences of observations collected from various aircraft sensors during the flight. A fault in the aircraft results in anomalous readings in sequences collected from one or more of the sensors. Similarly, in health-care domain, an abnormal medical condition in a patient's heart can be detected by identifying anomalies in the time-series corresponding to *Electrocardiogram* (ECG) recordings of the patient.

Traditional anomaly detection techniques assume that the data exists in the form of univariate or multivariate records, and ignore the sequential aspect of the data. Often, anomalies in sequences can be detected only by analyzing data instances together as a sequence, and hence cannot be detected by the traditional anomaly detection techniques. While the traditional anomaly detection problem has been widely researched [35, 84], the research in the area of anomaly detection for sequences is at a relatively preliminary stage. The objective of this thesis is to advance the state of art for the problem of anomaly detection for sequences.

Anomaly detection for sequences is a rich problem domain due to two major reasons. First, the nature of the sequence data varies across different application domains. Sequences can be *symbolic*, such as gene sequences, or sequence of system calls, or *time series*, such as ECGs, climate time series, stock prices, etc. Sequences can be *univariate*, in which each observation of a sequence is a univariate symbol or real value, or *multivariate*, in which each observation of a sequence is a multivariate vector consisting of symbolic, continuous, or heterogenous mixture of univariate observations. Some domains deal with complex sequences such as sequence of graphs, or sequence of documents, etc. Each type of sequence entails a unique set of challenges which are required to be handled in a unique manner. The second reason that contributes to the richness of the problem is that anomalies in sequences can be defined in multiple ways. For example, an entire sequence can be anomalous with respect to a set of normal sequences. Similarly, a subsequence within a long sequence can be anomalous with respect to the rest of the sequence. Each of the different definitions of anomalies are fundamentally distinct from each other and cannot be solved in the same manner.

1.1 Anomaly Detection for Sequences — State of Art

The problem of anomaly detection for sequences is not as well understood as the traditional anomaly detection problem. For certain types of sequences, like symbolic sequences, several anomaly detection techniques have been proposed [36], while for other types of sequences, such as univariate and multivariate time series, limited techniques exist. Existing research on anomaly detection for sequences has been fragmented across different application domains, without an understanding of how the performance of the techniques relates to the various aspects of the problem, such as nature of data, nature of anomalies, etc. Thus techniques that are shown to perform well in one domain are not guaranteed to perform well in a different domain, since the nature of the sequence data encountered in the two domains is often significantly different.

In this thesis, we focus on the problem of detecting if a given sequence is anomalous with respect to a database of normal sequences. We study the problem for three types of sequences — *univariate symbolic sequences*, *univariate time series*, and *multivariate time series*.

	Univariate Symbolic Sequences		Univariate Time Series		Multivariate Time Series	
Research Goals	**State of Art**	**Thesis Contrib.**	**State of Art**	**Thesis Contrib.**	**State of Art**	**Thesis Contrib.**
1. *Anomaly Detection Techniques?*	Several	√	Limited	√	None	√
2. *Cross-domain Understanding?*		√		√		
3. *Best Technique for a Domain?*		√				
4. *Best Overall Technique?*		√				

Table 1.1: Thesis Contributions : Bridging Gaps Between State of Art and the Research Goals

Table 1.1 illustrates the gap that exists between the current state of art and the desired goal for each sequence type. The ultimate research goal for any type of sequence data is to find an anomaly detection technique that can perform best on any data set. In Table 1.1, we list the intermediate research goals that need to be met, to reach the ultimate goal. Most of the current research in this area has been limited to techniques that have been proposed within specific application domains. For symbolic sequences, most of the existing techniques either focus on detecting anomalies in data obtained in the domain of system call intrusion detection or in the domain of biological sequences. For time series data, the research is even more sparse.

This thesis aims at bridging the gap between the current state of research on anomaly detection for sequences and the ultimate research goal, as illustrated in Table 1.1. To achieve the goal of developing an anomaly detection technique that can perform well on symbolic sequence data from any application domain, we solve the following problems in a step by step fashion:

1. *Propose novel anomaly detection techniques for sequences.* We propose several novel anomaly detection techniques for different types of sequences. In particular, we implement a window based strategy, in which a sequence is broken into fixed length subsequences or “windows”. Each window is treated as a unit of analysis. The window based techniques proposed in this thesis are shown to perform comparably or better than the existing techniques. This is especially significant for multivariate time series data, for which there are no existing anomaly detection techniques.

2. *Evaluate performance of techniques across domains.* An anomaly detection technique, when applied in two different application domains, might encounter two remarkably different types of data. Thus it is important to know how the performance of a given technique varies across domains. In this thesis, we conduct extensive experimental evaluation of the proposed anomaly detection techniques and several state of art techniques on data sets collected from a variety of application domains. The cross-domain evaluation provides an unbiased comparison of the techniques, which can be used by a domain scientist when choosing an anomaly detection technique for a particular application domain.

3. *Understand the strengths and weaknesses of different techniques.* A key observation from our experimental evaluations is that none of the techniques are found to be superior across all domains, but their performance is closely tied to the nature of the data they are applied to. We study the relationship between the performance of anomaly detection techniques and the nature of the sequence data using a formal analysis framework called *Reference Based Analysis* (RBA) framework. The essence of this framework is to characterize a data set using a set of real valued features, and then relate the different techniques to one or more of these features.

4. *Propose techniques that performs best across domains.* The real valued features obtained using RBA not only allow analysis of a data set but can also be used as features for anomaly detection. We propose a novel technique that utilizes RBA features which is shown to outperform all existing techniques for symbolic sequences across data sets obtained from a variety of application domains.

1.2 Contributions

This thesis makes the following key contributions:

1.2.1 A Reference Based Analysis Framework for Exploring Data

A novel analysis framework, *Reference Based Analysis* (RBA), is proposed that can map a given data set (of any type) into a multivariate continuous space with respect to a reference data set. The key property of the mapping is that the data instances which are similar to the reference data are mapped to a different region in the multivariate space than the instances that are different from the reference data. The RBA framework is used to not only visualize and understand complex data types, such as multivariate categorical data and symbolic sequence data, but also to extract data driven features which when used with traditional anomaly detection techniques are shown to consistently outperform the state of art anomaly detection techniques for these complex data types. Two novel techniques for symbolic sequences, WIN_{1D} and WIN_{2D} are proposed using the RBA framework.

1.2.2 Anomaly Detection Techniques for Symbolic Sequences

We provide an experimental evaluation of a large number of anomaly detection techniques for symbolic sequences on a variety of data sets. We propose a novel window based technique called $FSAz$ which consistently performs better than most of the existing techniques. The experimental analysis allows relative comparison of the different anomaly detection techniques and highlights their strengths and weaknesses. We propose a novel artificial data generator that can be used to generate validation data sets to evaluate anomaly detection techniques for sequences. The generator allows to generate data sets with different characteristics by varying the associated parameters to study the relationship between the anomaly detection techniques and the different characteristics of the data.

To understand the relationship between anomaly detection techniques and nature of sequence data, we use the RBA framework to characterize data, and then identify how the performance of each technique is related to one or more of these characteristics.

We propose two techniques, WIN_{1D} and WIN_{2D} using the RBA based features for symbolic sequences. These techniques are designed to better utilize the difference between normal and anomalous data, and are shown to perform better than the best anomaly detection techniques for each application domain.

1.2.3 Anomaly Detection Techniques for Univariate Time Series Data

We study the problem of detecting anomalies in a time series database. We investigate different ways of solving this problem. First way is to use distance or similarity kernels for time series data. Second is to extract fixed length windows from a test time series and assign anomaly scores to the windows. Third is to learn a predictive or forecasting model from the training data and use the model to detect anomalies in a test time series. Fourth is to learn a state space model for the normal time series and detect anomalies by passing a test time series through the state space model. We adapt several machine learning techniques (such as one class support vector machines, nearest neighbor density estimation, support vector regression) to detect anomalies in time series data. We evaluate these novel adaptations along with existing state of the art anomaly detection techniques for time series data [191, 32]. One of our novel adaptations, $WINC_{SVM}$, is shown to perform better than the existing anomaly detection techniques. To understand the performance of existing anomaly detection techniques for symbolic sequences [38], we discretized the continuous time series data and applied the symbolic techniques on the discretized data. We provide useful insights regarding the relative performance of different techniques based on the experimental evaluation and relate the performance of different techniques to the nature of the underlying time series data.

1.2.4 Anomaly Detection Techniques for Multivariate Time Series Data

We propose a novel window based anomaly detection technique, called WIN_{SS} for multivariate time series data. The proposed technique uses concept of *subspace monitoring* developed to reduce a multivariate time series into a univariate time series, capturing the dynamics of the data. The strength of WIN_{SS} is that it is first attempt to model both the sequence and the multivariate aspect of the data to detect anomalies. Experimental results show that WIN_{SS} is significantly better than techniques that either ignore the sequence aspect or the multivariate aspect of the data.

Software and Data Sources

Implementations for the anomaly detection techniques described in this thesis as well as the data sets used for the experimental evaluation are available at `http://www.cs.`

`umn.edu/~chandola/software.html` [33].

1.3 Thesis Outline

This thesis is organized in following three parts:

Part I provides the necessary background required for the rest of this thesis. Chapter 2 discusses the general problem of anomaly detection and provides a brief overview of the related work in area of anomaly detection. Chapter 3 discusses the problem of anomaly detection for symbolic sequences and time series data. We provide the different existing problem formulations and the related research. We also provide an exact definition of the anomaly detection problem solved by the techniques proposed in this thesis and the evaluation methodology used to measure the performance of the techniques. Chapter 4 introduces a novel data analysis framework, known as *Reference Based Analysis* (RBA), which is instrumental in understanding the performance of different anomaly detection techniques.

Part II deals with anomaly detection for univariate symbolic sequences. We provide a comparative evaluation of anomaly detection techniques for univariate symbolic sequences in Chapter 5. In Chapter 6, we show how a novel data analysis framework, called *Reference Based Analysis* (RBA), is applied to understand the performance of different anomaly detection techniques and to develop novel anomaly detection techniques for univariate symbolic sequences.

Part III deals with anomaly detection for time series data. We study anomaly detection techniques for univariate time series data in Chapter 7. In Chapter 8, we propose a novel technique to detect anomalies in multivariate time series. We conclude with future directions of research in Chapter 9.

Part I

Background

Chapter 2

Anomaly Detection — Background and Definitions

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior. These non-conforming patterns are often referred to as anomalies, outliers, discordant observations, exceptions, aberrations, surprises, peculiarities or contaminants in different application domains. Of these, anomalies and outliers are two terms used most commonly in the context of anomaly detection; sometimes interchangeably. Anomaly detection finds extensive use in a wide variety of applications such as fraud detection for credit cards, insurance or health care, intrusion detection for cyber-security, fault detection in safety critical systems, and military surveillance for enemy activities.

Traditionally, anomaly detection techniques treat data as a collection of multivariate records. A large and diverse literature on techniques that handle such data exists, and has been covered in several survey articles and books [35, 84, 5, 138, 161, 154, 13, 79, 17, 9]. The literature on anomaly detection techniques for sequence data is relatively sparse [36, 41].

The importance of anomaly detection is due to the fact that anomalies in data translate to significant (and often critical) actionable information in a wide variety of application domains. For example, an anomalous traffic pattern in a computer network could mean that a hacked computer is sending out sensitive data to an unauthorized destination [110]. An anomalous MRI image may indicate presence of malignant tumors [163]. Anomalies in credit card transaction data could indicate credit card or identity theft [6] or anomalous readings from a space craft sensor could signify a fault in some component of the space craft [64].

In this chapter we discuss the problem of anomaly detection and provide the related work done in this area. Section 2.1 provides a general definition of anomalies. A key aspect of the anomaly detection problem is the different ways in which anomalies can be defined as discussed in Section 2.2. We provide an overview of the related research done in the area of anomaly detection in Section 2.3.

2.1 What are anomalies?

Anomalies are patterns in data that do not conform to a well defined notion of normal behavior. Figure 2.1 illustrates anomalies in a simple 2-dimensional data set. The data has two normal regions, N_1 and N_2, since most observations lie in these two regions. Points that are sufficiently far away from the regions, e.g., points o_1 and o_2, and points in region O_3, are anomalies.

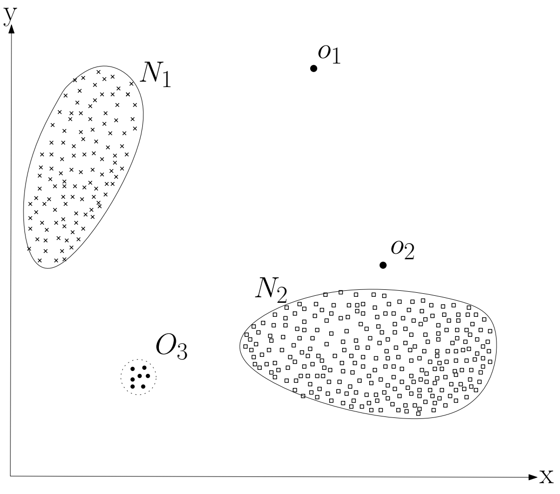


Figure 2.1: A simple example of anomalies in a 2-dimensional data set.

Anomalies might be induced in the data for a variety of reasons, such as malicious activity, e.g., credit card fraud, cyber-intrusion, terrorist activity or breakdown of a system, but all of the reasons have a common characteristic that they are *interesting* to the analyst. The "interestingness" or real life relevance of anomalies is a key feature of anomaly detection.

Anomaly detection is related to, but distinct from *noise removal* [178] and *noise accommodation* [154], both of which deal with unwanted *noise* in the data. Noise can be defined as a phenomenon in data which is not of interest to the analyst, but acts as a hindrance to data analysis. Noise removal is driven by the need to remove the unwanted objects before any data analysis is performed on the data. Noise accommodation refers to immunizing a statistical model estimation against anomalous observations [93].

Another topic related to anomaly detection is *novelty detection* [131, 132, 157] which aims at detecting previously unobserved (*emergent, novel*) patterns in the data, e.g., a new topic of discussion in a news group. The distinction between novel patterns and anomalies is that the novel patterns are typically incorporated into the normal model after being detected.

It should be noted that solutions for above mentioned related problems are often used for anomaly detection and vice-versa, and hence are discussed in this review as well.

2.2 Different Aspects of An Anomaly Detection Problem

This section identifies and discusses the different aspects of anomaly detection. As mentioned earlier, a specific formulation of the problem is determined by several different factors such as the nature of the input data, the availability (or unavailability) of labels as well as the constraints and requirements induced by the application domain. This section brings forth the richness in the problem domain and justifies the need for the broad spectrum of anomaly detection techniques.

2.2.1 Nature of Input Data

A key aspect of any anomaly detection technique is the nature of the input data. Input is generally a collection of data instances (also referred as *object, record, point, vector, pattern, event, case, sample, observation, entity*) [172, Chapter 2] . Each data instance can be described using a set of attributes (also referred to as *variable, characteristic, feature, field, dimension*). The attributes can be of different types such as *binary*, *categorical* or *continuous*. Each data instance might consist of only one attribute (*univariate*) or multiple attributes (*multivariate*). In the case of multivariate data instances, all attributes might be of same type or might be a mixture of different data types.

The nature of attributes determine the applicability of anomaly detection techniques. For example, for statistical techniques different statistical models have to be used for continuous and categorical data. Similarly, for nearest neighbor based techniques, the nature of attributes would determine the distance measure to be used. Often, instead of the actual data, the pairwise distance between instances might be provided in the form of a distance (or similarity) matrix. In such cases, techniques that require original data instances are not applicable, e.g., many statistical and classification based techniques.

Input data can also be categorized based on the relationship present among data instances [172]. Most of the existing anomaly detection techniques deal with record data (or point data), in which no relationship is assumed among the data instances.

In general, data instances can be related to each other. Some examples are *sequence data*, *spatial data*, and *graph data*. In sequence data, the data instances are linearly ordered, e.g., time-series data, genome sequences, protein sequences. In *spatial data*, each data instance is related to its neighboring instances, e.g., vehicular traffic data, ecological data. When the spatial data has a temporal (sequential) component it is referred to as *spatio-temporal* data, e.g., climate data. In *graph data*, data instances are represented as vertices in a graph and are connected to other vertices with edges. Later in this section we will discuss situations where such relationship among data instances become relevant for anomaly detection.

2.2.2 Type of Anomaly

An important aspect of an anomaly detection technique is the nature of the desired anomaly. Anomalies can be classified into following three categories:

Point Anomalies

If an individual data instance can be considered as anomalous with respect to the rest of data, then the instance is termed as a point anomaly. This is the simplest type of anomaly and is the focus of majority of research on anomaly detection.

For example, in Figure 2.1, points o_1 and o_2 as well as points in region O_3 lie outside the boundary of the normal regions, and hence are point anomalies since they are different from normal data points.

As a real life example, consider credit card fraud detection. Let the data set correspond to an individual's credit card transactions. For the sake of simplicity, let us assume that the data is defined using only one feature: *amount spent*. A transaction for which the amount spent is very high compared to the normal range of expenditure for that person will be a point anomaly.

Contextual Anomalies

If a data instance is anomalous in a specific context (but not otherwise), then it is termed as a contextual anomaly (also referred to as *conditional anomaly* [162]).

The notion of a context is induced by the structure in the data set and has to be specified as a part of the problem formulation. Each data instance is defined using following two sets of attributes:

1. *Contextual attributes.* The contextual attributes are used to determine the context (or neighborhood) for that instance. For example, in spatial data sets, the longitude and latitude of a location are the contextual attributes. In time-series data, time is a contextual attribute which determines the position of an instance on the entire sequence.

2. *Behavioral attributes.* The behavioral attributes define the non-contextual characteristics of an instance. For example, in a spatial data set describing the average rainfall of the entire world, the amount of rainfall at any location is a behavioral attribute.

The anomalous behavior is determined using the values for the behavioral attributes within a specific context. A data instance might be a contextual anomaly in a given context, but an identical data instance (in terms of behavioral attributes) could be considered normal in a different context. This property is key in identifying contextual and behavioral attributes for a contextual anomaly detection technique.

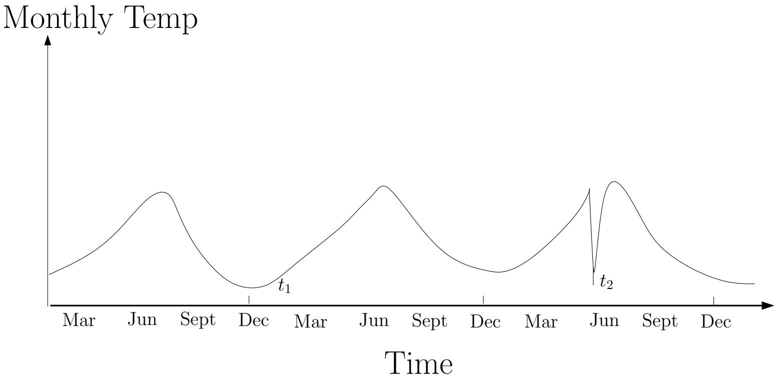


Figure 2.2: Contextual anomaly t_2 in a monthly temperature time series.

Contextual anomalies have been most commonly explored in time-series data [185, 155] and spatial data [108, 159]. Figure 2.2 shows one such example for a temperature time series which shows the monthly temperature of an area over last few years. A temperature of 35F might be normal during the winter (at time t_1) at that place, but the same value during summer (at time t_2) would be an anomaly.

A similar example can be found in the credit card fraud detection domain. A contextual attribute in credit card domain can be the *time* of purchase. Suppose an individual usually has a weekly shopping bill of \$100 except during the Christmas week, when it reaches \$1000. A new purchase of \$1000 in a week in July will be considered a contextual anomaly, since it does not conform to the normal behavior of the individual in the context of time (even though the same amount spent during Christmas week will be considered normal).

The choice of applying a contextual anomaly detection technique is determined by the meaningfulness of the contextual anomalies in the target application domain. Another key factor is the availability of *contextual* attributes. In several cases defining a context is straightforward, and hence applying a contextual anomaly detection technique makes sense. In other cases, defining a context is not easy, making it difficult to apply such techniques.

Collective Anomalies

If a collection of related data instances is anomalous with respect to the entire data set, it is termed as a collective anomaly. The individual data instances in a collective anomaly may not be anomalies by themselves, but their occurrence together as a collection is anomalous. Figure 2.3 illustrates an example which shows a human electrocardiogram output [72]. The highlighted region denotes an anomaly because the same low value exists for an abnormally long time (corresponding to an *Atrial Premature Contraction*). Note that that low value by itself is not an anomaly.

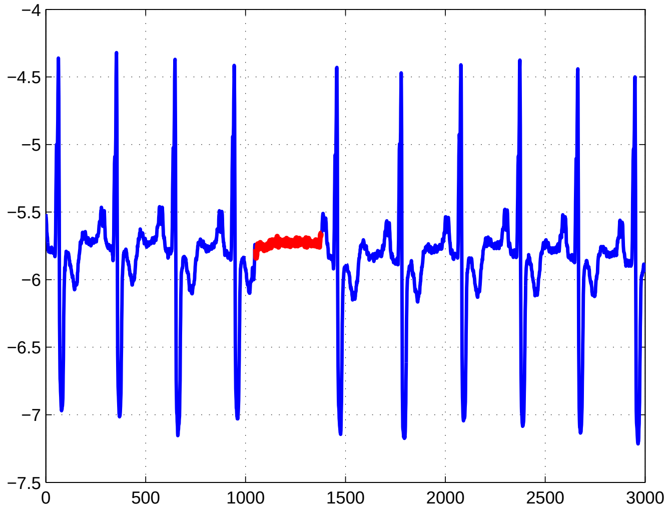


Figure 2.3: Collective anomaly corresponding to an *Atrial Premature Contraction* in an human electrocardiogram output.

As an another illustrative example, consider a sequence of actions occurring in a computer as shown below:

… http-web, buffer-overflow, http-web, http-web, smtp-mail, ftp, http-web, ssh, smtp-mail, http-web, **ssh**, **buffer-overflow**, **ftp**, http-web, ftp, smtp-mail,http-web …

The highlighted sequence of events (**buffer-overflow**, **ssh**, **ftp**) correspond to a typical web based attack by a remote machine followed by copying of data from the host computer to remote destination via *ftp*. It should be noted that this collection of events is an anomaly but the individual events are not anomalies when they occur in other locations in the sequence.

Collective anomalies have been explored for sequence data [59, 169], graph data [135], and spatial data [159].

It should be noted that while point anomalies can occur in any data set, collective anomalies can occur only in data sets in which data instances are related. In contrast, occurrence of contextual anomalies depends on the availability of context attributes in the data. A point anomaly or a collective anomaly can also be a contextual anomaly if analyzed with respect to a context. Thus a point anomaly detection problem or collective anomaly detection problem can be transformed to a contextual anomaly detection problem by incorporating the context information.

The techniques used for detecting collective anomalies are very different than the point and contextual anomaly detection techniques, and require a separate detailed discussion. Hence we chose to not cover them in this survey. For a brief review of the research done in the field of collective anomaly detection, the reader is referred to an extended version of this survey [34].

2.2.3 Data Labels

The labels associated with a data instance denote if that instance is *normal* or *anomalous*[1]. It should be noted that obtaining labeled data which is accurate as well as representative of all types of behaviors, is often prohibitively expensive. Labeling is often done manually by a human expert and hence requires substantial effort to obtain the labeled training data set. Typically, getting a labeled set of anomalous data instances which cover all possible type of anomalous behavior is more difficult than getting labels for normal behavior. Moreover, the anomalous behavior is often dynamic in nature, e.g., new types of anomalies might arise, for which there is no labeled training data. In certain cases, such as air traffic safety, anomalous instances would translate to catastrophic events, and hence will be very rare.

Based on the extent to which the labels are available, anomaly detection techniques can operate in one of the following three modes:

---

[1] Also referred to as normal and anomalous classes.

Supervised anomaly detection

Techniques trained in supervised mode assume the availability of a training data set which has labeled instances for normal as well as anomaly class. Typical approach in such cases is to build a predictive model for normal *vs.* anomaly classes. Any unseen data instance is compared against the model to determine which class it belongs to. There are two major issues that arise in supervised anomaly detection. First, the anomalous instances are far fewer compared to the normal instances in the training data. Issues that arise due to imbalanced class distributions have been addressed in the data mining and machine learning literature [96, 97, 40, 143, 186, 182]. Second, obtaining accurate and representative labels, especially for the anomaly class is usually challenging. A number of techniques have been proposed that inject artificial anomalies in a normal data set to obtain a labeled training data set [179, 1, 166]. Other than these two issues, the supervised anomaly detection problem is similar to building predictive models. Hence we will not address this category of techniques in this survey.

Semi-Supervised anomaly detection

Techniques that operate in a semi-supervised mode, assume that the training data has labeled instances for only the normal class. Since they do not require labels for the anomaly class, they are more widely applicable than supervised techniques. For example, in space craft fault detection [64], an anomaly scenario would signify an accident, which is not easy to model. The typical approach used in such techniques is to build a model for the class corresponding to normal behavior, and use the model to identify anomalies in the test data.

A limited set of anomaly detection techniques exist that assume availability of only the anomaly instances for training [46, 45, 58]. Such techniques are not commonly used, primarily because it is difficult to obtain a training data set which covers every possible anomalous behavior that can occur in the data.

Unsupervised anomaly detection

Techniques that operate in unsupervised mode do not require training data, and thus are most widely applicable. The techniques in this category make the implicit assumption that normal instances are far more frequent than anomalies in the test data. If this assumption is not true then such techniques suffer from high false alarm rate.

Many semi-supervised techniques can be adapted to operate in an unsupervised mode by using a sample of the unlabeled data set as training data. Such adaptation assumes that the test data contains very few anomalies and the model learnt during training is robust to these few anomalies.

2.2.4 Output of Anomaly Detection

An important aspect for any anomaly detection technique is the manner in which the anomalies are reported. Typically, the outputs produced by anomaly detection techniques are one of the following two types:

Scores

Scoring techniques assign an anomaly score to each instance in the test data depending on the degree to which that instance is considered an anomaly. Thus the output of such techniques is a ranked list of anomalies. An analyst may choose to either analyze top few anomalies or use a cut-off threshold to select the anomalies.

Labels

Techniques in this category assign a label (*normal* or *anomalous*) to each test instance. Several techniques, internally, calculate a score for each test instance and use either a threshold or a statistical test to assign a label.

Scoring based anomaly detection techniques allow the analyst to use a domain-specific threshold to select the most relevant anomalies. Techniques that provide binary labels to the test instances do not directly allow the analysts to make such a choice, though this can be controlled indirectly through parameter choices within each technique.

2.3 Related Work for Traditional Anomaly Detection

In this section we provide a brief overview[2] of the related research in the area of anomaly detection. Anomaly detection techniques can be grouped into following broad categories:

- **Classification based techniques** learn a classifier from a labeled (or unlabeled) training data and assign an anomaly score or label to a test data instance [177, 175, 176, 11, 153, 80, 126, 127].

- **Nearest neighbor based techniques** analyze the nearest neighborhood of a test instance to assign it an anomaly score [149, 106, 107, 136, 174, 26, 25].

- **Clustering based techniques** learn clusters from a given data set and assign an anomaly score to a test instance based on its relationship with its nearest cluster [51, 82, 130, 51, 145, 129].

- **Statistical techniques** estimate a parameteric or non-parametric model from the data and apply a statistical test on the probability of the instance to be generated by the estimated model to assign an anomaly score to the test instance [13, 60, 2, 113, 43].

- **Spectral decomposition based techniques** find an approximation of the data using a combination of attributes that capture the bulk of variability in the data. Instances that are significantly different from others in the lower approximation are detected as anomalies [3, 137, 160, 64, 148].

- **Information theoretic techniques** analyze the *information content* of a data set using different information theoretic measures such as *Kolomogorov Complexity, entropy, relative entropy*, etc and detect instance that induce irregularities in the information content of the data set as anomalies [7, 103, 117, 83, 81].

- **Contextual anomaly detection techniques** analyze a context around each test data instance to determine if the instance is anomalous or not. Contextual anomaly detection techniques have been developed to handle spatial data [123, 159, 108, 168] and sequence data [2, 60, 180, 65, 196, 94, 182, 186].

---

[2] The reader is referred to [35] for an extended survey on anomaly detection techniques.

Chapter 3

Anomaly Detection for Sequences – Background and Related Work

In this chapter we will discuss the problem of anomaly detection for sequence data. Anomaly detection for sequences is fundamentally distinct from the traditional point anomaly detection problem discussed in Chapter 2. A key characteristic of the problem is that it can be defined in multiple ways such that each problem definition is unique. In this chapter we will discuss different problem formulations that relevant for symbolic sequences and time series data. We will also provide a brief overview of the existing research for each of these problem formulation.

As mentioned in Chapter 2, a large number of anomaly detection techniques exist for point anomaly detection, which typically deal with multivariate non-sequential data. While such techniques, when applied to sequences, can detect individual observations that are anomalous (i.e. point anomalies), they cannot detect contextual and collective anomalies.

For example, consider the set of user command sequences shown in Table 3.1. Clearly the sequence S_5 is anomalous, corresponding to a hacker breaking into a computer after multiple failed attempts, even though each command in the sequence by itself is normal. Thus traditional techniques, which analyze each data instance (an individual command), cannot detect such anomalies.

S_1	*login, pwd, mail, ssh, . . . , mail, web, logout*
S_2	*login, pwd, mail, web, . . . , web, web, web, logout*
S_3	*login, pwd, mail, ssh, . . . , mail, web, web, logout*
S_4	*login, pwd, web, mail, ssh, . . . , web, mail, logout*
S_5	**login**, **pwd**, **login**, **pwd**, **login**, **pwd**, . . . , **logout**

Table 3.1: Sequences of User Commands

For time series data, we refer the reader to Figure 2.3 in Chapter 2, which corresponds to a human electrocardiogram output [72]. The highlighted region denotes an anomaly because the same low value exists for an abnormally long time (corresponding to an *Atrial Premature Contraction*). Note that that low value by itself is not an anomaly and hence the anomalous region cannot be detected by traditional techniques that do not account for the sequential nature of the data.

The rest of this chapter is organized as follows. Section 3.1 describes three different anomaly detection problem formulations for sequences. We elaborate on semi-supervised anomaly detection problem formulation in 3.2, since the techniques proposed in this thesis solve that specific problem. We provide the evaluation methodology that we adopt to evaluate the semi-supervised anomaly detection techniques in Section 3.3. Section 3.4 provides an overview of existing research on anomaly detection for symbolic sequences. Section 3.5 provides an overview of existing research on anomaly detection for univariate time series data. Section 3.6 provides an overview of existing research on anomaly detection for multivariate time series data.

3.1 Problem Formulations

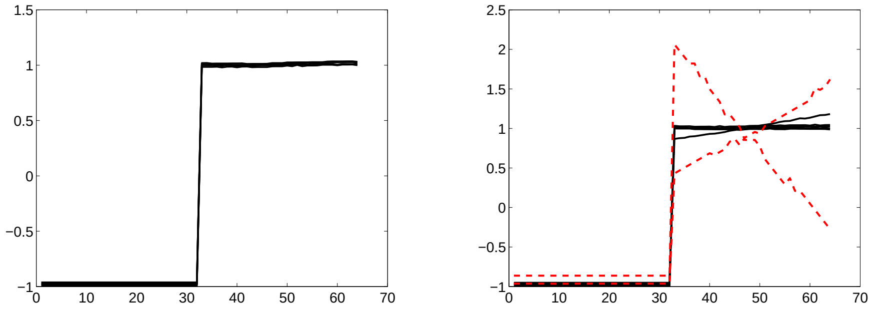
For sequence data, the problem of anomaly detection can be formulated in following three distinct ways:

3.1.1 Detecting Anomalous Sequences w.r.t a Sequence Database

The first problem formulation for sequences is to determine if a given test sequence is anomalous with respect to a database of sequences. The training database is assumed to consist of mostly normal sequences. Thus this problem formulation is similar to the semi-supervised point anomaly detection problem, only difference being that data points are replaced with sequences. A variant of the semi-supervised problem formulation is the case when the sequence database is unlabeled and can contain both normal as well as anomalous sequences. Thus the problem formulation is to detect all sequences in a given database which are anomalous, and is similar to the unsupervised point anomaly detection problem.

For example, consider the following scenario that can arise in the domain of operating system intrusion detection. A security analyst is interested in detecting "illegal" user sessions on a computer belonging to a corporate network. An illegal user session is caused when an unauthorized person uses the computer with malicious intent. To detect such intrusions, the analyst can use the first formulation, in which the past normal user sessions (sequence of system calls/commands) are used as the training data, and a new user session is tested against this training data.

For time series data, consider the example shown in Figure 3.1. Figure 3.1(a) shows a set of reference time series corresponding to measurements from a healthy rotary engine disk of an aircraft [165], and Figure 3.1(b) shows a test set of time series corresponding to measurements from healthy (solid) and cracked (dashed) disks.This task requires assigning an anomaly score (or label) to the test time series.



(a) Reference

(b) Test

Figure 3.1: Normal and test time series for aircraft engine data.

3.1.2 Detecting Anomalous Subsequence within a Sequence

The second problem formulation for sequences is to detect any subsequence within a given long sequence which is anomalous with respect to the rest of the sequence. This problem formulation is also referred to as *discord detection*, in the context of time series [101].

For example, consider the following scenario in the domain of system call intrusion detection. A security analyst is interested in detecting if a user's account was misused (hacked) in the past few months. To detect this misuse, the analyst can use the second formulation, in which the user's activity for the past few months is considered as a long sequence, and is tested for any anomalous subsequence.

For time series data, Figure 2.3 shows an example of a discord, which can be detected using the second problem formulation. For time series data, a special type of sequence anomaly occurs when the length of the anomalous subsequence is 1, i.e., one is interested in finding individual observations that are anomalous in the time series; these individual observations are also referred to as *outliers*. A vast literature in the time series statistics community exists that deals with outlier detection in time series data. See Section 3.5 for more discussion on such techniques.

3.1.3 Determining if the frequency of a query pattern in a given sequence is anomalous w.r.t its expected frequency

The third problem formulation for sequences is to detect if the frequency of occurrence a given short query subsequence in a long test sequence is anomalous with respect to its frequency of occurrence in a database of normal sequences. This problem formulation is relevant only for symbolic sequences, since the frequency of exact occurrence of a continuous subsequence is not significant.

For symbolic sequences, this problem is also referred to as *surprise detection* [102, 120]. Going back to the example from system call intrusion detection given in Table 3.1, the sequence **login,passwd,login,passwd** corresponds to a failed login attempt followed by a successful login attempt. Occurrence of this sequence in a user's daily profile is normal if it occurs occasionally, but is anomalous if it occurs very frequently, since it could correspond to an unauthorized user surreptitiously attempting an entry into the user's computer by trying multiple passwords. To detect such intrusions, the analyst can use the third formulation, in which the sequence of commands is the query pattern, and the frequency of the query pattern in the user sequence for the given day is compared against the expected frequency of the query pattern in the daily sequences for the user in the past, to detect anomalous behavior.

3.2 Semi-supervised Anomaly Detection for Sequences

In this thesis, we address a specific anomaly detection problem formulation for sequence data. The key objective here is to detect anomalous sequences with respect to a database of normal sequences. The exact formulation of the problem addressed by techniques in this thesis is:

Problem 1 *Given a set of* n *training sequences,* $\mathbf{T}$*, and another set of* m *test sequences* $\mathbf{S}$*, find the anomaly score* $A(S_q)$ *for each test sequence* $S_q \in \mathbf{S}$*, with respect to* $\mathbf{T}$*.*

The length of sequences in $\mathbf{S}$ and sequences in $\mathbf{T}$ might or might not be equal in length. The training database $\mathbf{T}$ is assumed to contain mostly normal sequences, and hence will also be referred to as normal database.

Several anomaly detection techniques for sequences, solve the following unsupervised version of Problem 1:

Problem 2 *Given a set of* n *sequences,* $\mathbf{S}$*, find the anomaly score* $A(S_q)$ *for each sequence* $S_q \in \mathbf{S}$*.*

Problem 1 can be posed as Problem 2, and vice-versa. In this thesis, to maintain uniformity, we have adapted all techniques to solve Problem 1.

3.3 Evaluation Methodology

Evaluating anomaly detection techniques is a critical part of the anomaly detection research. Since most of the techniques are developed in the context of a particular application domain, it is essential to assess how well does a technique perform in terms of detecting anomalies from a given test data set. Additionally, an evaluation metric is required to compare anomaly detection techniques and find the best technique suited for a particular problem. Evaluation of anomaly detection techniques is challenging because often a labeled validation data set is not available. Here we discuss certain aspects regarding evaluation of anomaly detection techniques for sequences. Another challenge arises since the techniques are scoring based, and do not provide a binary classification of test sequences as normal or anomalous. Thus traditional classification based evaluation metrics, such as accuracy, precision, and recall, cannot be directly applied.

3.3.1 Lack of Availability of Labeled Validation Data

In most domains, since anomalies correspond to a rare, and often improbable, event, a validation data set that contains labeled anomalies and normal sequences is not always available. In such cases, one has to rely on evaluation methods that do not require knowledge of a labeled validation data. One such method is to use a validation set consisting of only normal sequences (which is relatively easier to obtain). A strong anomaly detection technique will assign low scores to all validation data instances, while the scores assigned by a weak technique will have a wider “spread”. Note that this evaluation only assesses how well a technique can detect normal sequences from a test set, but does not guarantee if the technique will assign higher scores to anomalous sequences.

If an unlabeled validation set containing both normal and anomalous sequences is available, and the proportion of normal and anomalous sequences is known, one can visualize the distribution of anomaly scores assigned by a technique. If the distribution is shows *bi-modality*, and the relative sizes of the two “modes” are same as the known proportion, the anomaly detection technique might be performing better than a technique for which the distribution does not exhibit any bi-modality. Again, such behavior does not guarantee the performance of the technique, but can be used as an indicator to compare techniques, or select suitable parameter settings.

3.3.2 Evaluating Scoring Based Techniques

Evaluation of an anomaly detection technique, when a labeled validation set is available, is still challenging. The output of a typical anomaly detection technique is an anomaly score for every test instance. While this score can be used to rank the instances based on their degree of anomaly, it cannot be directly converted into a “classification type” binary output.

Threshold Based Evaluation To convert the anomaly scores into a binary output, a threshold on the anomaly score can be employed, such that any score above the threshold is labeled as anomalous and any score below the threshold is labeled as normal. Alternatively, one can label top few ranked instances (according to the anomaly score) as anomalous and rest as normal. In this thesis, we use the latter method to evaluate the output of any anomaly detection technique as follows:

1. Rank the test sequences in decreasing order based on the anomaly scores.
2. Label the sequences in the top p portion of the sorted test sequences as anomalous, and rest sequences as normal, where $1 < p \leq s$, where s is the total number of test sequences.

The binary output can now be used to measure standard metrics such as precision, recall, misclassification accuracy, etc [172]. We evaluate the technique using the following evaluation metric:

$$Accuracy = \frac{t}{p} \tag{3.1}$$

where t is the true anomalous sequences in top p ranked sequences. The metric measures how precise is the anomaly detection technique for the top p ranked anomalies. Thus if an analyst has resources to process only the top p ranked anomalies, the metric measures how many ”true” anomalies will be processed. In this thesis, we report the accuracy when $p = q$, where q is number of true anomalous sequences in the test data set.

Area Under ROC One drawback of the above threshold based metric is that it is highly dependent on the choice of the threshold p. A technique might show 100% accuracy for a particular value of p but show 50% accuracy for $2p$. An alternative evaluation metric that can be computed for an anomaly detection technique is the *Area Under ROC*[1] (AUC). The ROC can be constructed by varying the threshold p from 1 to

---

[1] Receiver Operating Characteristic[53].

s and computing the false positive rate and the true positive rate for each value of p. A technique with higher AUC is better than a technique with lower AUC. An advantage of using AUC is that it is independent of any threshold. A possible disadvantage of using AUC is that it measures the behavior of the technique for the entire output, while often, an analyst might only be interested in top few ranked anomalies. Thus even if one technique might be better than another based on AUC, but the former might not detect as many ”true” anomalies in the top ranks.

3.3.3 Other Metrics

While the above two metrics are generic for any type of data, additional metrics are sometimes required to evaluate anomaly detection techniques for sequences. One metric is the *localization of anomaly* and measures how accurately can a technique detect the location(s) of anomaly(ies) in a test sequence. Certain techniques can only determine if a given sequence is anomalous or not, without any information about the location of the anomalous region(s). This metric is important for domains such as system health monitoring, to help diagnose the problem in a system, such as an aircraft or spacecraft, that causes the anomaly. Another metric is *time to detect* the anomaly in a sequence and measures how quickly can a technique detect an anomaly in a test sequence, after the real anomaly has started. This metric is important when the testing is being done in an online fashion, such as monitoring health of a spacecraft in operation. A technique that can detect the occurrence of an anomaly as soon as it actually occurs can be used for real time monitoring, while a technique that can detect anomalies only after observing the entire sequence are not useful in such scenarios.

In this thesis, we evaluate the anomaly detection techniques using the threshold based accuracy metric and the AUC metric. We also evaluate the scalability of the techniques based on the time taken by the technique for training and testing.

3.4 Anomaly Detection for Symbolic Sequences – Related Work

In this section, we provide an overview of the existing research on anomaly detection for symbolic sequences. Bulk of research in this area focusses on the first problem formulation as discussed in Section 3.1.1. Such techniques can be grouped into following categories:

- **Kernel Based Techniques:** These techniques treat the entire test sequence as a unit element in the analysis [28, 29, 189], and hence are analogous to point based anomaly detection techniques. They typically apply a proximity based point anomaly detection technique by defining an appropriate similarity kernel for the sequences.
- **Window Based Techniques:** These techniques analyze a short window of symbols—a short subsequence—within the test sequence at a time [58, 87, 50, 48, 70, 69, 112, 111, 30]. Thus such techniques treat a subsequence within the test sequence as a unit element for analysis. These techniques require an additional step in which the anomalous nature of the entire test sequence is determined, based on the analysis on the subsequences within the entire sequence.
- **Markovian Techniques:** These techniques predict the probability of observing each symbol of the test sequence, using a probabilistic model, and use the per-symbol probabilities to obtain an anomaly score for the test sequence [169, 192, 133, 52, 116]. These techniques analyze each symbol with respect to previous few symbols.
- **Hidden Markov Model Based Techniques:** These techniques transform the input sequences into sequences of hidden states, and then detect anomalies in the transformed sequences [59, 147, 198, 57].

Though several techniques have been proposed for symbolic sequences in various application domains, there has not been any cross domain evaluation and understanding of the existing techniques. Forrest et al [59] compared four different anomaly detection techniques, but evaluated them in the context of system call intrusion detection. Sun et al [169] proposed a technique for protein sequences, but no evaluation with techniques proposed for system call data was done. Similarly, while Budalakoti et al [28] proposed a clustering based techniques to detect anomalies in flight sequences, it has not been shown how the same technique would perform on system call intrusion detection data or protein data.

Most of the anomaly detection techniques that handle the second problem formulation (Section 3.1.2) slide a fixed length window across the given long sequence and compare each window with the remaining sequence to detect anomalous windows [101, 100, 119, 184].

Techniques that handle the third problem formulation (Section 3.1.3) compute the frequency of the query pattern in normal sequences and in the test sequence and compare the frequencies to assign an anomaly score to the query pattern [102, 120, 77, 78].

3.5 Anomaly Detection for Univariate Time Series Data – Related Work

In this section, we provide an overview of the existing research on anomaly detection univariate time series data. Several statistical techniques detect anomalous observations (also referred to as *outliers*) within a single time series using various time series modeling techniques such as Regression [60, 2, 154], Auto Regression (AR) [63, 188], ARMA [144], ARIMA [195], Support Vector Regression (SVR) [124], Kalman Filters [105], etc. The general approach behind such techniques is to forecast the next observation in the time series, using the statistical model and the time series observed so far, and compare the forecasted observation with the actual observation to determine if an anomaly has occurred.

Two broad categories of techniques have been proposed to solve the first problem formulation (Section 3.1.1, viz., segmentation based and kernel based anomaly detection techniques. The general approach behind segmentation based techniques is to segment the normal time series, and treat each segment as a state in a *Finite State Automaton* (FSA), and then use the FSA to determine if a test time series is anomalous or not. Several variants of the segmentation based technique have been proposed [32, 128, 156]. Kernel based anomaly detection techniques compute similarity/distance between time series and apply a nearest neighbor based anomaly detection technique on the similarity "kernel" [146, 183, 191]. Protopapas et al [146] use *cross correlation* as the similarity measure, and compute the anomaly score of a test time series as the inverse of its average similarity to all other time series in the given data set. Wei et al [183] use a *rotation invariant* variant of Euclidean distance to compute distance between time series, and then assign an anomaly score to each time series as equal to its distance to its nearest neighbor. Yankov et al [191] proposed pruning based heuristics to improve the efficiency of the nearest neighbor technique [183].

Several anomaly detection techniques for time series data solve the second problem formulation (Section 3.1.2, also referred to as *discord detection* [101, 100, 103, 119, 62, 27, 191]. Such techniques analyze fixed length windows obtained from the time series by comparing each window with the rest of the time series or against all other windows from that time series. A window which is significantly different from other windows is declared as a discord.

Similar to symbolic sequences, the existing techniques for time series data have been proposed within application domains and have not been compared with techniques proposed for other domains. For example, while Mahoney and Chan [32] propose a segmentation based technique to detect anomalies in data obtained from spacecraft, it has not been compared with techniques that have been proposed for ECG data, nor has it been applied to other domains to test its performance.

3.6 Anomaly Detection for Multivariate Time Series Data – Related Work

Limited research has been done to solve the semi-supervised anomaly detection problem for multivariate time series data. Most of the existing anomaly detection techniques for multivariate time series focus on detecting a single anomalous multivariate observation [10, 66, 181]. Baragona and Battaglia [10] propose an ICA based technique to detect outliers in multivariate time series. The underlying idea is to isolate the multivariate time series into a set of independent univariate components and an outlier signal, and analyze the univariate outlier signal to determine the outliers. The ICA based technique assumes that the observed signals are linear combination of independent components as well as independent noise signal, and the the added noise has a high kurtosis.

Cheng et al [42] proposed a distance based approach to detects anomalous subsequences within a given multivariate sequence. For a given multivariate sequence S, all w length windows are extracted. The distance between each pair of windows is computed to obtain a symmetric $(T - w + 1) \times (T - w + 1)$ kernel matrix. A fully connected graph

	Type of Sequence Data		
	Univariate Symbolic	Univariate Time Series	Multivariate Time Series
System Call Intrusion Detection	[58, 87, 51, 52, 112]		
Network Intrusion Detection	[51]		
Aircraft/Space Shuttle Safety	[28, 29]	[165, 63, 155, 156, 32, 128]	[64, 90]
Cardiac/Brain Health Monitoring		[184, 100, 103]	[10]
Eco-system Disturbance Detection		[22]	[42]
Proteomics	[76, 169, 122]		
Fraud Detection	[31, 140, 139]		
Astronomy		[146, 150, 191]	

Table 3.2: Applications of Anomaly Detection for Sequences

is constructed using the kernel matrix in which each node represents a w length window and the weight on the edges between the pair of windows is equal to the similarity (inverse of distance) between the pair. The nodes (or components) of the graph that have least connectivity are declared as anomalies.

3.7 Applications of Anomaly Detection for Sequences

Anomaly detection is a highly application oriented field of research. Most of the existing techniques for sequences have been proposed in the context of different application domains. We summarize the different application domains in which anomaly detection techniques for different types of sequences have been applied in Table 3.2.

Chapter 4

A Reference Based Analysis Framework for Data

A key, though often overlooked, aspect of data mining based research is the relationship between the performance of an algorithm and the nature of the data to which the algorithm is applied. This is even more significant in the context of anomaly detection, since it is an application oriented field of research. An anomaly detection algorithm, when applied in two different application domains, might encounter two remarkably different types of data. Thus it is essential to understand the characteristics of a given data set and to relate the performance of an anomaly detection algorithm to these characteristics, to understand how a given algorithm will perform on a given data set. In this chapter we introduce a novel analysis framework to characterize a data set. In Chapter 6, we will show how the framework can be used to understate the performance of a given anomaly detection algorithm.

The *Reference Based Analysis* (RBA) framework can be used to analyze a given data set, with respect to a reference data set. This framework is motivated from the way semi-supervised anomaly detection techniques differentiate between normal and anomalous test instances by learning a model of normal behavior from a normal training set. The strength of the RBA framework is that it can be used to analyze complex types of data, for which limited analysis techniques exist. The key feature of the RBA framework is its ability to analyze a given data set with respect to a reference data set. In the transformed space, unseen instances similar to the reference data set will tend to occupy the same region that is occupied by instances from the reference data set. By contrast, instances that are different (we call this the *novel* class) will tend to be mapped to other regions, at least for some of the dimensions. This transformation of categorical data to continuous space can be utilized in practice for a variety of purposes.

The rest of this chapter is organized as follows. Section 4.1 provides a motivation for RBA framework from the perspective of semi-supervised anomaly detection. Section 4.2 provides a general introduction to categorical data and the motivation for the RBA framework for such type of data. Section 4.3 provides the related work done in the field of analyzing multivariate categorical data. Section 4.4 defines a set of separability statistics which form the core of the RBA framework. Section 4.5 describes how the separability statistics can be used to map categorical data into a continuous space. Sections 4.6 and 4.7 describe how the RBA framework can be employed for visualization and anomaly detection for categorical data, respectively. We conclude with potential future extensions of the RBA framework in Section 4.8.

4.1 Motivation for Reference Based Analysis Framework

Semi supervised anomaly detection techniques distinguish between normal and anomalous instances in a given test data set, using a reference (or training) data set consisting of mostly normal instances. Each technique relies on the relationship between a test instance and reference data set and an assumption that the relationship between a normal test instance and the reference set is different than the relationship between an anomalous test instance and the reference set. We refer to such relationship as a *distinguishing characteristic* of the test instances.

For example, the k-nearest neighbor (kNN) based anomaly detection technique [149] assigns an anomaly score to each test instance equal to its distance to its k^{th} nearest neighbor in the reference set. Thus the *distance to the* k^{th} *nearest neighbor* in the reference set is the distinguishing characteristic for the kNN technique. Similarly, a standard clustering based anomaly detection technique assigns an anomaly score to each test instance equal to its distance to the centroid of the closest cluster from the reference set. Thus the *distance to the closest cluster* in the reference set is the distinguishing characteristic for the clustering based technique.

The key aspect of the distinguishing characteristics is that they are computed for every test instance using the reference set. It is straightforward to note that for kNN to perform well on a given data set, the normal and anomalous test instances need to be different from each other in terms of the distinguishing characteristic. Thus the distinguishing characteristics for a given test set are indicative of how well the corresponding anomaly detection techniques will perform on the test set. In other words, if the test data is mapped along a single dimension that represents a distinguishing characteristic, then the corresponding anomaly detection technique will perform well if the data is *separable* along this dimension.

The RBA framework is motivated from such distinguishing characteristics. For any type of data, a set of canonical features (also referred to as *separability statistics*) are constructed, which are similar in spirit to the distinguishing characteristics, but are more fundamental and can be used to compute the characteristics. The test data is mapped into the multi-dimensional space constructed using the separability statistics. If a test set is separable along a dimension, then any technique that utilizes that dimension, will perform well on the test set, and vice-versa. One can also use the features to visualize the data, which is one important use of RBA framework, especially for data types for which no visualization schemes exist. Often, a test set might not be separable along a single dimension, but might be separable using a linear or non-linear combination of the dimensions. Thus RBA allows for devising novel anomaly detection techniques which utilize such combinations and are best suited for a given test set.

4.2 Categorical Data

Categorical data (also known as nominal or qualitative multi-state data) has become increasingly common in modern real-world applications. Table 4.1 shows a sample of a categorical data set. These data sets are often rich in information and are frequently encountered in domains where large-scale data sets are common, e.g., in network intrusion detection. However, unlike continuous data, categorical data attribute values cannot be naturally mapped on to a scale, making most continuous data analysis techniques inapplicable in this setting: Table 4.2 lists common exploratory analysis techniques for continuous data and categorical data. As one can see, many techniques that are applicable to continuous data have no natural analogues in the categorical space.

cap-shape	cap-surface	···	habitat	Class
convex	smooth		urban	poisonous
convex	smooth		grasses	edible
bell	smooth		meadows	edible
convex	scaly		urban	poisonous
convex	smooth		grasses	edible
...				

Table 4.1: Sample of the Mushroom Data Set from the UCI Machine Learning Repository [8].

When exploring the characteristics of a multi-dimensional continuous data set, we might begin by looking at one attribute at a time. We could compute the mean, percentiles, variance and skewness, or construct a box plot, histogram or nonparametric density function. This would give us an idea of the range and overall distribution of each attribute. However, with categorical data we can only look at the mode or an unordered histogram. With ordinal data (ordered categorical data), we may also be able to look at percentiles but for the most part the situation is similar to categorical data.

Other techniques that are extremely valuable in exploring continuous data including factor analysis techniques such as PCA, or multidimensional scaling can give us an idea about the variability of the data across all attributes. Multivariate techniques such as these are not even applicable in the categorical setting. Regardless of our final goal in analyzing a continuous data set, all of the above steps would help us understand its characteristics. On the other hand, when given a categorical data set many of these exploratory steps cannot naturally be extended to the new setting, leaving a huge "gap" as can be seen from Table 4.2. Thus, there is a need for elemental approaches for exploring the characteristics of a categorical data set.

In this chapter we propose the RBA framework in the context of multivariate categorical data [37]. We seek to utilize underlying data characteristics for categorical data analysis, in the spirit of data-driven similarity measures. Specifically, we introduce the concept of *separability statistics*, which characterize the differences between a given

	Continuous	Categorical
Single Attribute	Mean, Median, Box Plot, Histogram, Percentile, Variance, Skewness, Density Function	Mode, Histogram (no ordering)
Pairs of Attributes	Covariance, Scatter Plot, Correlation, 2-D Histogram, Density Function	Contingency Table, Correspondence Analysis, 2-D Histogram (no ordering)
Entire Space	PCA, Subspaces, MDS, LLE, SVD, ISOMAP, FastMAP	Subspaces, Data Cube
Other Techniques	Correlation Matrix[1], LDA	Correlation Matrix[1], Discriminant Correspondence Analysis

Table 4.2: Exploratory data analysis techniques for continuous and categorical data.

instance and a labeled reference data set. Each statistic essentially represents a distance between an instance and the reference data set (i.e., the statistic allows mapping of the categorical data into a 1-dimensional continuous space). Therefore, using these statistics and a reference data set, one can map any collection of categorical instances (including those from the reference data set) to a multidimensional continuous space.

Clustering and outlier detection require a similarity measure when applied to categorical data. In previous work [24], we have shown that the choice of similarity measure significantly affects overall performance. The proposed framework provides the capability to define a better similarity measure for a particular categorical data set; we will demonstrate this in the context of anomaly detection, although one can extend this to other data mining tasks such as classification as well.

To illustrate the utility of separability statistics, let us consider a simple example. The *Mushroom* Data Set is a well-known categorical data set available from the UCI Machine Learning Repository [8]. This data set has 22 categorical attributes describing the various characteristics of a mushroom and a class which denotes whether a mushroom is edible or poisonous; the number of values taken by each of the attributes ranges between 2 and 12. While one can always explore the data set using techniques in Table 4.2 such as an unordered histogram, these techniques are limited in what they can reveal about the *joint* distribution of the attributes. Table 4.1 shows the first few data instances in the Mushroom data set over a subset of the attributes. Using the methods to be discussed in this paper, this data set was mapped to a continuous space for visualization. Figure 4.1 shows the data instances in this transformed space with markers defined by the true labels: it is evident that the classes are well separated in this space. This allows the analyst to visually explore the classes in the Mushroom data set, which is not easy to do for the original categorical data set.

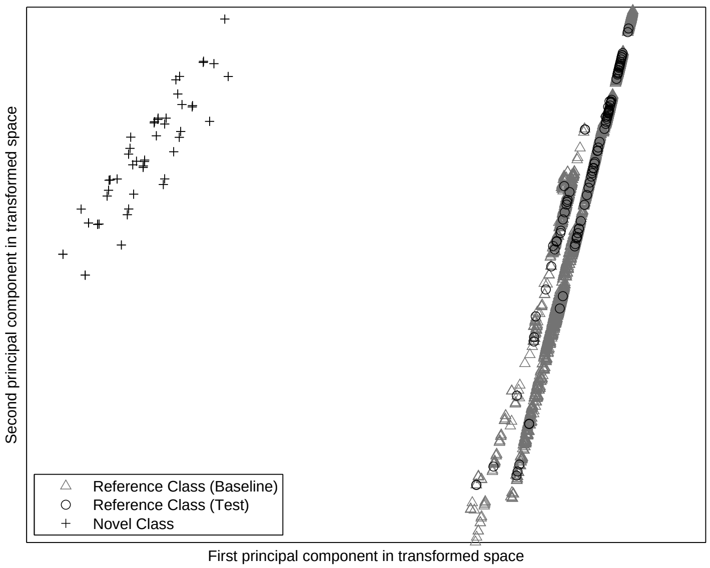


Figure 4.1: Visualization of the *Mushroom* data set using the proposed framework.

4.3 Related Work

Data with categorical attributes has been studied for a very long time, dating back at least a century when Karl Pearson [141, 142] introduced the χ^2 test for independence between categorical attributes. The traditional exploratory techniques used are contingency tables, the chi-square statistic, unordered histograms and pie charts [4]. Friendly [61] proposed several sophisticated statistical techniques such as *Sieve Diagrams* and *Mosaic Displays* to view k-way contingency tables, and *Multiple Correspondence Analysis* (MCA), to handle multivariate categorical data sets, though most techniques are limited to attributes that take few possible values. Fernandez [56] discusses several exploratory techniques for categorical data from a data mining perspective.

There have been a number of studies directed at categorical data in the visualization

---

[1] A matrix which shows the intra- and inter-class correlation in a block structure [173, chap. 3].

community [18, 21, 85, 86]. In particular, one direction in visualization has been to order the categories using the information present in the data [20, 125]. One such technique, called *Distance Quantification Classing* (DQC), was proposed by Rosario et al [152] to order the categories present in a class variable in a categorical data set with respect to the predictor variables. None of the techniques directly address the problem of analyzing a categorical data set with respect to a reference data set, which is the focus of our paper.

A number of unsupervised learning algorithms have been proposed for categorical data, e.g. CLICKS [194], CLOPE [190], ROCK [75], CACTUS [67], COOLCAT [12], and other techniques [71, 92]. Most of these techniques use some notion of similarity when comparing instances. Similarity measures that are devised using the framework proposed in this paper can be plugged in to many such algorithms.

Several probabilistic analysis techniques have such as Naive Bayes and other Bayesian techniques, Linear Discriminant Analysis, etc., are applicable for modeling categorical data. Such techniques are different from RBA since they operate on a single labeled data set, while the RBA framework analyzes data with respect to a reference data set. While the objective of the probabilistic modeling methods is to estimate the probabilities of observing test instances, the objective of RBA is to map categorical data into a multidimensional continuous space.

While RBA allows one to map categorical data into a multi-dimensional continuous space, one alternative method to achieve similar mapping is to *binarize* the categorical data. Thus each attribute, which takes k values, gets transformed into k binary values. A data instance with a particular value for that attribute has 1 at the corresponding binary location and 0 for all other locations. In this chapter we will show how RBA is significantly better than *binarization* for categorical data due to two reasons, first that binarization results in a large number of features, especially when the categorical attributes take a large number of possible values, and second that binarization does not utilize the notion of the reference set while transforming the data instances into a continuous space.

4.4 Separability Statistics

In this section we present a set of data-driven separability statistics that can be calculated for a given test data set with respect to a reference data set. Each statistic allows mapping of the categorical data into a 1-dimensional continuous space. The statistics are meant to differentiate instances in the reference data set from instances in other data sets. Since, for categorical data, the difference can be characterized in many different ways, a variety of separability statistics are possible. We only consider a few in this chapter.

(a) Data set.

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_2	d_1
a_1	b_1	c_3	d_1
a_2	b_1	c_4	d_2
a_2	b_1	c_5	d_1
a_1	b_2	c_6	d_3
a_1	b_2	c_7	d_3
a_2	b_2	c_8	d_3
a_2	b_2	c_9	d_3
a_2	b_2	c_{10}	d_4

(b) Characteristics of attributes and values.

arity	A: 2	B: 2	C: 10	D: 4
frequency	a_1: 5	b_1: 5	c_1: 1	d_1: 4
	a_2: 5	b_2: 5	c_2: 1	d_2: 1
			c_3: 1	d_3: 4
			c_4: 1	d_4: 1
			c_5: 1	
			c_6: 1	
			c_7: 1	
			c_8: 1	
			c_9: 1	
			c_{10}: 1	

Table 4.3: A simple categorical data set with four attributes.

The discussion of the separability statistics is organized in the following manner: we will begin by discussing the intuition behind each of the statistics, including motivating examples, and then proceed to formally define the statistics. For now, let us refer to the four statistics as d_m, f_m, n_x and f_x. Let us also consider the simple categorical data set shown in Table 4.3, and the following two data instances: $\mathbf{y} = \langle a_1, b_1, c_{10}, d_1 \rangle$ and $\mathbf{z} = \langle a_3, b_2, c_{10}, d_5 \rangle$.

The statistic d_m essentially captures the extent to which a given instance has matching values with instances in the reference data set. This is driven by the intuition that an instance belonging to the same class as the reference class will, on average, have more matching values with the reference class than an instance belonging to a different class. The procedure to map categorical data to continuous space will be discussed in

Section 4.5. For the purposes of the example being discussed with instances **y** and **z**, a brief outline is as follows: each statistic is computed by comparing a given instance with every instance in the reference data set, and then taking the average. For the instance **y**, the values corresponding to the first and last rows of the data set would be 3 and 1 respectively. The final value of this statistic for the instances **y** and **z** is 1.5 and 0.6, respectively.

The statistic f_m takes into account the *frequency* of matching values between an instance and reference data set. One way to think of this statistic is as a frequency-weighted version of the statistic d_m. The key intuition here is that in addition to the importance of more matching values, instances belonging to the reference class will also tend to match on relatively frequent values, while instances not belonging to the reference class will tend to match on infrequent values. This is important in situations where an attribute in the reference data set takes a very large number of values (e.g. the IP address in a network intrusion data set) thus making the odds of any match high. The value of this statistic for the instances **y** and **z** is 6.7 and 2.6, respectively.

The statistic n_x is a function of the *arity* of the mismatching attributes between an instance and a reference data set. In particular, the value of the statistic is higher when the mismatching attributes have lower arity, i.e. they take fewer values. The idea is that if an instance mismatches on an attribute that takes very few values across many instances in the reference class, then it is unlikely that it belongs to the class (simply because there are few opportunities to not match). The value of this statistic for the instances **y** and **z** is -5.45 and -7.90, respectively.

The statistic f_x looks at the frequency of mismatching attribute values between an instance and a reference data set. In a sense, this statistic is the "complement" of the f_m statistic and the intuition is also related to n_x; if the frequency of mismatching values is high between a given instance and most members of the reference class is high, then this means the instance often mismatches with the reference class on values that are common in the reference class. Thus, it is unlikely that the instance belongs to the same class as the reference class. The value of this statistic for the instances **y** and **z** is -1.57 and -2.725, respectively.

The values assigned by the four statistics for instances **y** and **z** suggest that **y** belongs to the reference class and **z** does not. This is somewhat difficult to conclude just by looking at the instances and the reference data set, but by examining the underlying quantities behind the statistics one can see that it is indeed reasonable to say that **y** and **z** belong to different classes. In particular, we have seen how the statistics map an instance between categorical space and continuous space based on several key underlying characteristics of the data set.

4.4.1 Formal Definition

Table 4.4 lists the notation that will be used in the subsequent discussions.

T	Reference data set
D	Test data set
N	Size of reference data set
d	Number of attributes in T and D
a_i	i^{th} attribute $(1 \leq i \leq d)$
$\mathcal{A}_i$	Set of categorical values taken by a_i in T
n_i	Number of values taken by a_i $(= \|\mathcal{A}_i\|)$
$f_i(x)$	Number of times a_i takes value x in T

Table 4.4: Notation used for explaining RBA.

Given a pair of categorical data instances $z \in D$ and $y \in T$, we define a partitioning of attribute set $\mathcal{A}$ into $\mathcal{A}_m$ and $\mathcal{A}_x$, such that, $z_i = y_i, \forall i \in \mathcal{A}_m$ and $z_i \neq y_i, \forall i \in \mathcal{A}_x$. $\mathcal{A}_m$ denotes the set of matching attributes and $\mathcal{A}_x$ denotes the set of mismatching attributes for the pair z, y.

We compute the following quantities for the pair z, y:

$$d_m = |\mathcal{A}_m| \tag{4.1}$$

$$f_m = \sum_{i \in \mathcal{A}_m} f_{z_i} \tag{4.2}$$

$$n_x = -\sum_{i \in \mathcal{A}_x} \frac{1}{n_i} \tag{4.3}$$

$$f_x = -\sum_{i \in \mathcal{A}_x} (\frac{1}{z_i} + \frac{1}{y_i}) \tag{4.4}$$

Thus, for every pair of categorical data instances $z \in D$ and $y \in T$ we have the following 4-tuple: $\langle d_m, f_m, n_x, f_x \rangle_{zy}$. For a test instance z we get a $|T| \times 4$ matrix of the above mentioned 4-tuple, denoted as:

$$\mathcal{M}_z = [\langle d_m, f_m, n_x, f_x \rangle_{zy}]_{\forall y \in T} \tag{4.5}$$

Let $\vec{z}_k$ denote a row vector containing top k^{th} value for each column of $\mathcal{M}_z$, such that:

$$\vec{z}_k = \langle d_{mk}, f_{mk}, n_{xk}, f_{xk} \rangle \tag{4.6}$$

For a given value of k, we define a set of 4 statistics denoted as the row vector $\vec{z}_k$.

The reason to choose the top k^{th} value from each column of $\mathcal{M}_z$ instead of a parameter independent value, such as the mean of the column, is to avoid issues due to multiple modes existing in the reference data set, T. If a very small value of k, such as 1, is chosen, the statistics can get affected by the presence of outliers in T. We have empirically observed that $5 \leq k \leq 15$ is a reasonable value of k for a variety of data sets. A set of statistics can be defined using multiple values of k to reduce the sensitivity on k.

Each of the four statistics mentioned in (4.6) are motivated from the following observations in context of two instances $z_1, z_2 \in D$ and $y \in T$, such that z_1 is similar to instances (generated by the same distribution as T) in T while z_2 is different from the instances in T (not generated by the same distribution as T):

1. $d_{m|z_1y} > d_{m|z_2y}$.
2. $f_{m|z_1y} > f_{m|z_2y}$.
3. $f_{x|z_1y} > f_{x|z_2y}$.
4. $n_{x|z_1y} > n_{x|z_2y}$.

The above mentioned arguments indicate that if test instances $z \in D$ are transformed or mapped to $\vec{z}_k$, then the instances similar to T will map to the same region, while the instances different from T will map to a different region.

It should be noted that all of the above four observations might not necessarily hold true at the same time for a given data set. But one or more of them will likely hold true and hence by mapping the data into the joint space, one can distinguish between the two types of test instances.

Measure	$S_i(z_i, y_i)$	$\propto$
Overlap	$= \begin{cases} 1 & \text{if } z_i = y_i \\ 0 & \text{otherwise} \end{cases}$	d_{mk}
Goodall	$= \begin{cases} \frac{f_i(z_i)(f_i(z_i)-1)}{N(N-1)} & \text{if } z_i = y_i \\ 0 & \text{otherwise} \end{cases}$	f_{mk}
OF	$= \begin{cases} 1 & \text{if } z_i = y_i \\ \frac{1}{1+\log \frac{N}{f_i(z_i)} \times \log \frac{N}{f_i(y_i)}} & \text{otherwise} \end{cases}$	d_{mk}, f_{xk}
Eskin	$= \begin{cases} 1 & \text{if } z_i = y_i \\ \frac{n_i^2}{n_i^2+2} & \text{otherwise} \end{cases}$	d_{mk}, n_{xk}

Table 4.5: Similarity Measures for Categorical Attributes. Note that $S(z, y) = \sum_{i=1}^{d} S_i(z_i, y_i)$.

4.4.2 Relationship to Similarity Measures

There have been several data driven similarity measures proposed for categorical data sets [24]. Table 4.5 lists four popular similarity measures that have been defined to measure similarity $S(z, y)$, between a pair of data instances.

We argue that the similarity of a test instance z to its k^{th} nearest neighbor in T using a data driven similarity measure, can be expressed as a function of one or more of the canonical statistics listed in (4.6). Column 3 in Table 4.5 indicates the particular test statistic that corresponds to each similarity measure.

As an illustrative example, consider the similarity measure *Goodall* listed in Table 4.5. Let us consider a test instance z and the reference data set T. The *Goodall* similarity of z with an instance $y \in T$ can be written as:

$$\begin{aligned} S(z, y) &= \sum_{i \in \mathcal{A}_m} \frac{f_i(z_i)(f_i(z_i) - 1)}{N(N-1)} + \sum_{i \in \mathcal{A}_x} 0 \\ &= \frac{1}{N(N-1)} \Big(\sum_{i \in \mathcal{A}_m} f_i(z_i)^2 - \sum_{i \in \mathcal{A}_m} f_i(z_i) \Big) \\ &\approx \frac{1}{N(N-1)} (f_m^2 - f_m) \end{aligned}$$

where $\mathcal{A}_m$ and $\mathcal{A}_x$ denote the set of attributes in which z and y match and mismatch, respectively. Note that we approximate $\sum_{i \in \mathcal{A}_m} f_i(z_i)^2$ with $(\sum_{i \in \mathcal{A}_m} f_i(z_i))^2$.

The similarity of z to its k^{th} nearest neighbor in T, using the *Goodall*, is equal to the k^{th} largest value of $S(z, y) \forall y \in T$, and can be written as:

$$S^k(z, y) \approx \frac{1}{N(N-1)}(f_{mk}^2 - f_{mk})$$

Thus we have shown how the *Goodall* similarity measure is related to the separability statistic f_{mk}. Similar relations can be shown for other similarity measures.

It may be argued that any similarity measure defined for categorical instances (such as the ones listed in Table 4.5 and others discussed in [24]) maybe used as a potential separability statistic in addition to the ones listed in (4.6). But the statistics proposed in this chapter are canonical and the similarity measures can be viewed as functions of one or more of the proposed statistics.

4.5 Mapping Data to Continuous Space

In this section we describe the process of mapping categorical data into a continuous space using the separability statistics discussed in Section 4.4.

For each categorical test instance in D, we first obtain the corresponding separability statistics as shown in (4.6) with respect to the reference set T, using one or more values for k. The characteristic of this mapping is that test instances that belong to the reference class have lower values for each statistic than test instances that belong to the novel class. We denote the mapped test data set with $\vec{D}$.

The reference data set T can also be mapped into a continuous space with respect to itself in the same manner as described above. We denote the mapped reference data set with $\vec{T}$. If the instances in T belong to a few dominant modes, one would expect the reference instances to map to similar values for each of the separability statistics.

Before further processing of the mapped data sets $\vec{D}$ and $\vec{T}$, it is desirable to normalize the data, since the different statistics can take different ranges of values. Each column of the mapped data set $\vec{D}$ is z-normalized to bring all statistics to the same scale. The mapped training data set $\vec{T}$ is also normalized but in a slightly different manner; the difference being that the z-normalization of each column in $\vec{T}$ is done using the column means and standard deviations obtained from the mapped test data set.

This is important, because if the z-normalization of $\vec{T}$ is done with respect to itself, the reference instances might have different normalized values for the statistics than the similar instances in the test set, which is not desirable.

Figure 4.2 highlights the significance of the normalization, as described above, using the *Mushroom* data set. The plot 4.2(a) shows the mapped data using the raw statistics f_{mk} and n_{xk}. The range of f_{mk} statistic is [2.0e+04,5.5e+04] while the range of n_{xk} statistic is [-0.53,-0.16]. If the reference and test data sets are normalized independently, the reference instances are mapped to different values than the test instances belonging to the reference class as can be seen in Figure 4.2(b). If both reference and test data sets are normalized with respect to the test data set, the reference instances are normalized in the same fashion as the test instances belonging to the reference class, as can be seen in Figure 4.2(c) which is a scaled down version of the raw data in Figure 4.2(a).

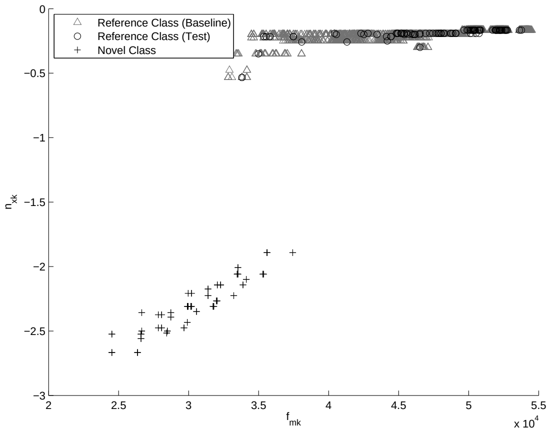
4.6 Visualization

The separability statistics described in Section 4.4 allows for the visual exploration of any categorical data set. The data is first transformed as discussed in Section 4.5. Since the resulting data space is continuous, it is suitable for visualization. In particular, it allows the analyst to visually explore aspects such as separation between modes, size and the number of modes.

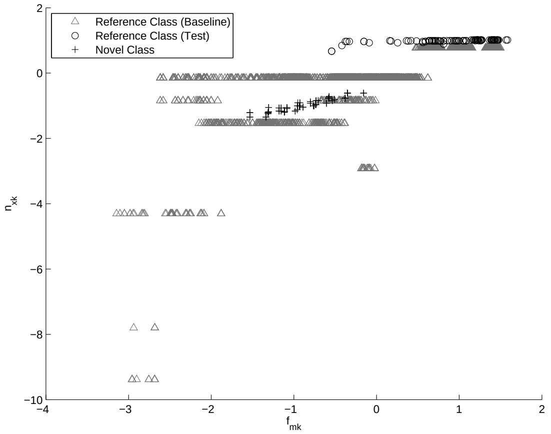
There are multiple ways to visualize the transformed continuous space, the simplest of which involve looking at pairs of dimensions or projections along specific subsets. Other mechanisms can be used to visualize continuous space such as tours in the GGobi system [171] and those in the Orca system [170]. We refer the reader to the recent work by Wickham et al [187] and Lawrence et al [114] for a discussion of high-dimensional data visualization systems. In this paper, we will discuss two ways, one which utilizes dimensionality reduction and another with histograms.

4.6.1 Two-dimensional Scatter Plots.

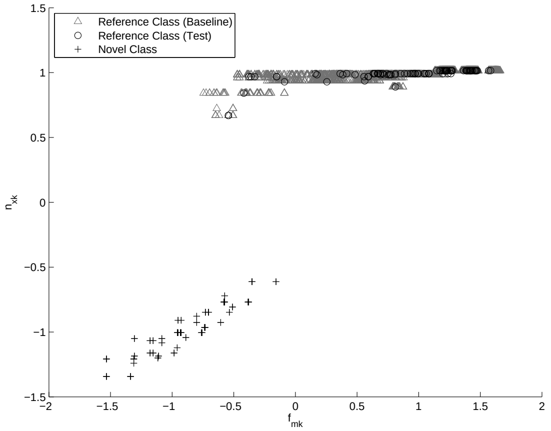
In order to reduce the number of dimensions to two for the purpose of visualization, we will use the well-known principal components analysis (PCA) technique [49]. The role of PCA here is to give more emphasis to the statistics that exhibit more separability



(a) No normalization.



(b) Reference and test data sets independently normalized.



(c) Reference and test data sets normalized with respect to the test data set.

Figure 4.2: Plots of *Mushroom1* data set using statistics f_{mk} and f_{xk}.

and filter out those that are close to unimodal. For this paper we have chosen PCA for its simplicity, however, there are other dimensionality reduction techniques that may be better suited for this task; we will not discuss other techniques since they are out of the scope of this paper.

To illustrate the use of our proposed framework for visualization, we will consider an example with a real data set. The data set has been partitioned into a labeled reference data set and a test data set. The test data set is then mapped to continuous space using the procedure discussed in Section 4.5. The resulting space $\vec{T}$ is 4-dimensional and each column is normalized; PCA is applied to the data set and the leading two principal components are preserved. The data set $\vec{T}$ is then projected on to the two leading principal components resulting in a 2-dimensional data set, with the number of rows being the number of test instances.

We can now visually explore the two-dimensional space; in this case we will use a scatter plot. The idea is that instances that have similar values for the statistics will end up in the same region of the plot, while those that have different values will be in different regions. The key observation is that the instances that have different values are likely to be from a different class than the reference data set.

Figure 4.3 shows the scatter plot for the KDD1 data set, which has 29 attributes, some of which take hundreds of values. Note that the labels of the test data set were examined only *after* the data was mapped to this space. The test data set contained instances from the reference class as well as instances that did not belong to the reference class. It is evident that the separability statistics were effective in distinguishing the classes. In particular, note that the instances belonging to the reference class were mapped to the same region, even though some of these instances came from the test class for which the label was unknown during the analysis. Another advantage of a dimensionality reduction technique is that it returns a linear combination of the statistics which is optimal in some sense. The weights from the linear combination can then be used to design a similarity measure for the data set (this aspect will be further discussed in Section 4.7.1).

As mentioned in Section 4.3, binarization is an alternative way to transform categorical data into a continuous space. Figure 4.4 shows the a scatter plot for first two principal components of the binarized KDD1 data set. We note that the visualization

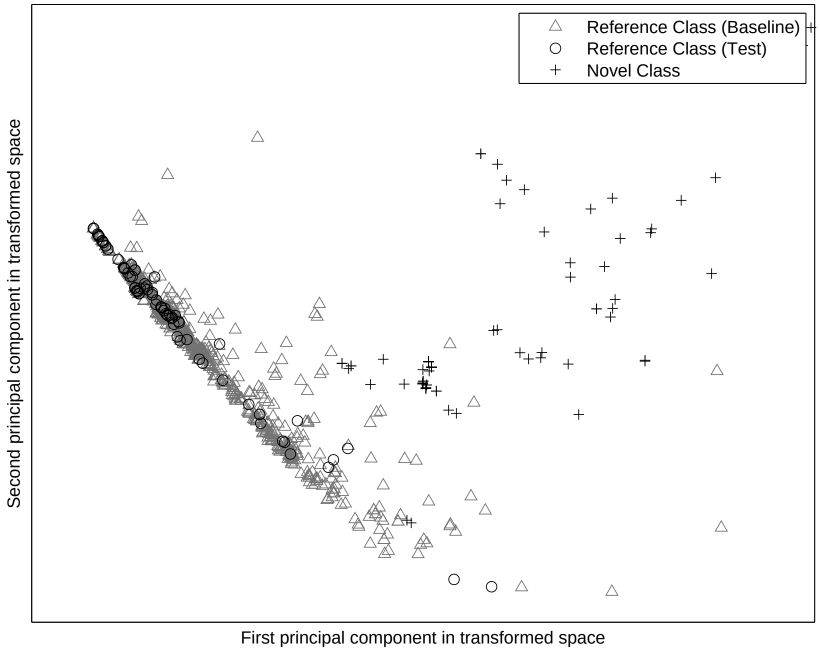


Figure 4.3: Visualization of the KDD1 data set mapped into RBA feature space.

obtained from binarized data does not reveal as insightful information regarding the original categorical data, when compared to the mapped data using RBA features.

4.6.2 Histograms

Since the separability statistics are directly capturing important characteristics of the underlying data, it is very useful to examine their distribution using a histogram. In this section, we will discuss exploring the distribution of a single statistic. As stated earlier, if a statistic assigns different values to a set of instances compared to the reference class, they are likely to be from a different class than the reference data set. Therefore, the distribution of the statistic will be unimodal (with low variance) when all instances are from the same class. One way to examine the distribution of a statistic is using a histogram. The histogram will essentially show to what extent the distribution departs from a low variance unimodal behavior.

Figure 4.5 shows histograms of the four statistics for the KDD1 data set (the labels were examined only after the histogram was constructed). In this case, it is apparent

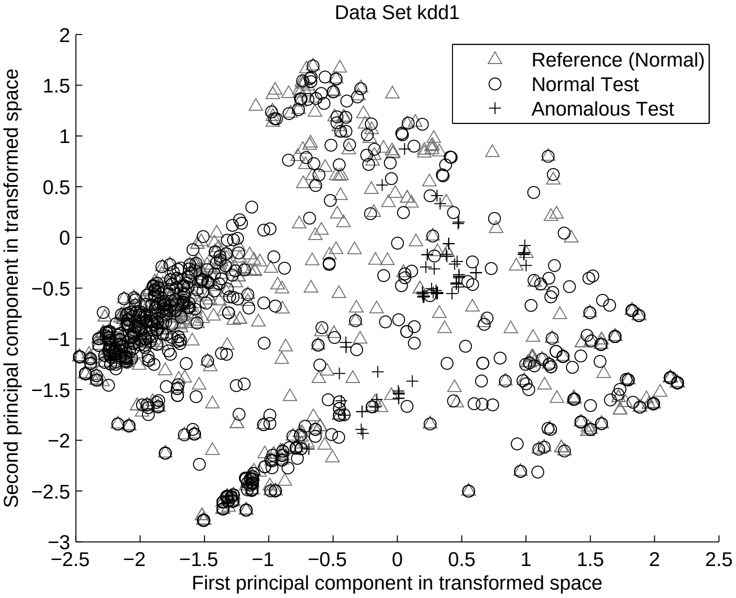
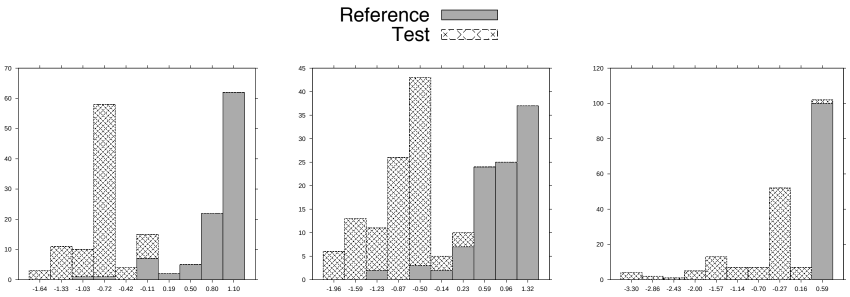


Figure 4.4: Visualization of the KDD1 data set mapped into binarized feature space.

that the distribution of all the statistics are multi-modal with high variance. If we were to generate this plot without knowing the labels, we would observe that the f_m and d_m statistics exhibited a high degree of multi-modality, while the other two statistics were somewhat multi-modal. Therefore, f_m and d_m would be considered the best separating statistics for this data set. Looking at Figure 4.5, taking the labels into account we see that this is indeed the case. Based on these histograms, the conclusion for the KDD1 data set would be that the reference class can be distinguished from other classes using properties related to the d_m statistic (more matching values) and the f_m statistic (more matches on frequent values).

4.7 Utility of Separability Statistics for Anomaly Detection

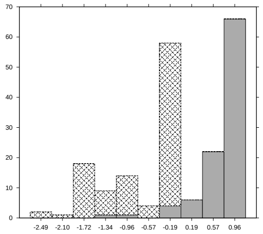
In this section we illustrate the utility of the separability statistics in semi-supervised anomaly detection. Here the objective is to separate anomalies from normal instances in a given test data set, with respect to a reference (training) data set which is assumed



(a) d_m statistic.

(b) f_m statistic.

(c) n_x statistic.



(d) f_x statistic.

Figure 4.5: Visualization of KDD1 data set using histograms.

to contain only normal instances.

We use a nearest neighbor based anomaly detection technique (kNN) [149, 173] which assigns the anomaly score of a test instance as equal to the distance of the test instance to its k^{th} nearest neighbor in the reference data set. The distance can be computed using any distance measure. If a measure computes similarity, the anomaly score of a test instance is inverse of the similarity to its k^{th} nearest neighbor.

We experimented with two kNN based anomaly detection techniques using the separability statistics. In the first variation (denoted as $kNNEuc$), we assign an anomaly score to each test instance in $\vec{D}$ using $\vec{T}$ as the reference data, using *Euclidean* distance as the distance measure.

In the second variation of kNN (denoted as $kNNPCA$), we use *Principal Component Analysis* (PCA) to project the mapped data sets, $\vec{D}$ and $\vec{T}$, to a lower dimensional space. PCA is performed on the mapped test data set $\vec{D}$. The top principal components that capture 90% of the variance in the test data are chosen. Both test and reference data sets are projected along these top principal components. Anomaly scores are assigned to test instances using *Euclidean* distance in this projected space. Both

	cr1	cr2	cn1	cn2	kd1	kd2	kd3	kd4	sk1	sk2	ms1	ms2	cen	bal	ttt	aud
d	6	6	42	42	29	29	29	29	10	10	21	21	10	4	9	16
$\|T\|$	904	944	3055	3055	1000	1000	6007	6007	2182	1429	3208	2916	2120	106	316	73
$\|D\|$	759	715	1100	550	1100	1100	1100	1100	1298	1177	1100	1100	2321	308	341	77

Table 4.6: Description of public data sets used for experimental evaluation. Each test data sets contains normal and anomalous data instances in ratio 10:1.

	cr1	cr2	cn1	cn2	kd1	kd2	kd3	kd4	sk1	sk2	ms1	ms2	cen	bal	ttt	aud	*Avg*
ovr	0.16	0.06	0.38	0.14	0.88	0.97	0.90	0.90	0.68	0.44	1.00	0.96	0.11	0.04	0.23	0.43	0.52
gd4	0.45	0.65	0.10	0.06	0.79	0.93	0.90	0.90	0.12	0.08	0.78	0.93	0.07	0.07	0.52	0.29	0.48
of	0.54	0.58	0.64	0.16	0.82	0.94	0.85	0.78	0.68	0.42	1.00	0.96	0.19	0.04	0.29	0.43	0.58
esk	0.51	0.54	0.39	0.14	0.88	0.96	0.90	0.90	0.68	0.30	1.00	0.96	0.23	0.04	0.23	0.43	0.57
iof	0.14	0.46	0.51	0.16	0.70	0.87	0.73	0.81	0.25	0.17	1.00	0.95	0.09	0.07	0.87	0.29	0.51
lin	0.00	0.00	0.29	0.26	0.86	0.96	0.90	0.88	0.75	0.60	1.00	0.97	0.09	0.21	0.45	0.29	0.53
lin1	0.42	0.65	0.28	0.24	0.91	0.95	0.82	0.09	0.72	0.39	1.00	0.97	0.18	0.00	0.23	0.29	0.51
gd1	0.00	0.00	0.20	0.22	0.81	0.90	0.00	0.01	0.69	0.30	1.00	0.81	0.12	0.25	0.35	0.43	0.38
gd2	0.54	0.71	0.62	0.22	0.78	0.89	0.18	0.11	0.69	0.55	1.00	0.96	0.16	0.04	0.32	0.43	0.51
gd3	0.01	0.00	0.24	0.18	0.81	0.91	0.00	0.11	0.69	0.41	1.00	0.96	0.16	0.14	0.32	0.43	0.40
smv	0.00	0.00	0.07	0.16	0.00	0.00	0.00	0.00	0.34	0.07	0.00	0.00	0.07	0.21	0.35	0.00	0.08
gmb	0.57	0.68	0.67	0.24	0.72	0.91	0.79	0.85	0.20	0.20	1.00	0.90	0.15	0.04	0.35	0.43	0.54
brb	0.12	0.52	0.43	0.14	0.91	0.96	0.90	0.90	0.66	0.36	1.00	0.96	0.10	0.18	0.87	0.29	0.58
anb	0.00	0.02	0.15	0.14	0.58	0.78	0.69	0.22	0.51	0.09	1.00	0.88	0.21	0.14	0.39	0.29	0.38
euc	0.55	0.65	0.18	0.14	0.89	0.96	0.90	0.90	0.66	0.26	1.00	0.96	0.18	0.11	0.35	0.71	0.59
pca	0.55	0.72	0.18	0.14	0.90	0.96	0.90	0.90	0.71	0.42	1.00	0.95	0.18	0.11	0.39	0.71	0.61
stt	0.54	0.65	0.38	0.16	0.91	0.98	0.90	0.90	0.71	0.73	1.00	0.96	0.18	0.14	0.45	0.71	0.64
	f_{xk}	f_{mk}	d_{mk}	f_{mk}	f_{xk}	f_{xk}	d_{mk}	d_{mk}	f_{xk}	f_{xk}	d_{mk}	d_{mk}	f_{xk}	f_{mk}	f_{mk}	d_{mk}	
Avg	0.30	0.40	0.34	0.17	0.77	0.87	0.66	0.60	0.57	0.34	0.93	0.88	0.15	0.11	0.41	0.40	

Table 4.7: Performance of similarity measures and separability statistics on public data sets using kNN ($k = 10$).

variations combine the four statistics when computing distance between instances.

The motivation behind using PCA is that the statistics that can discriminate between normal and anomalous data instances in the test data tend to have higher variance than the statistics that do not discriminate between normal and anomalous data instances. By using PCA, we can capture the statistics with greater discriminative power.

To evaluate the performance of any technique, we count the number of true anomalies in the top n portion of the sorted anomaly scores of the test instances, where n is the number of actual anomalies. Let o be the number of actual anomalies in the top p predicted anomalies. The accuracy of the algorithm is measured as $\frac{o}{n}$.

We compare the two variants described above with 14 different categorical similarity measures on several publicly available data sets. Four of these similarity measures are listed in Table 4.5. The other ten measures have been developed in different contexts, and have been evaluated in [24]. The details of the data sets are summarized in Table 4.6. Fourteen of these data sets are based on the data sets available at the UCI Machine Learning Repository [8], while two are based on network data generated by SKAION Corporation for the ARDA information assurance program [89]. Nine of these data sets were purely categorical while seven (kd1,kd2,kd3,kd4,sk1,sk2,cen) had a mix of continuous and categorical attributes. Continuous variables were discretized using the MDL method [54]. Another possible way to handle a mixture of attributes is to compute the similarity for continuous and categorical attributes separately, and then do a weighted aggregation. In this study we converted the continuous attributes to categorical to simplify comparative evaluation.

For each test data set there is a corresponding normal reference data set, and a labeled test data set. The results are summarized in Table 4.7. The row *stt* denotes the performance of kNN when using the best separability statistic as the only attribute. The best statistic is indicated in the last row. We make several observations from the results in Table 4.7.

The performance of the similarity measures depends on the data set, which is expected, since the measures are data-driven. This also indicates that the ability of the underlying statistic to distinguish between normal and anomalous data instances depends on the data set. Since each similarity measure is a function of one statistic, we observe that the similarity measure which uses the best statistic for a given data set, is generally the best performer.

The performance of $kNNEuc$ technique (using all separability statistics) is one of the best on average. This result shows that when all statistics are used together, the performance can often be better than using them individually, though in several cases the performance deteriorates considerably when all statistics are used (such as for $cn2$ and $sk2$).

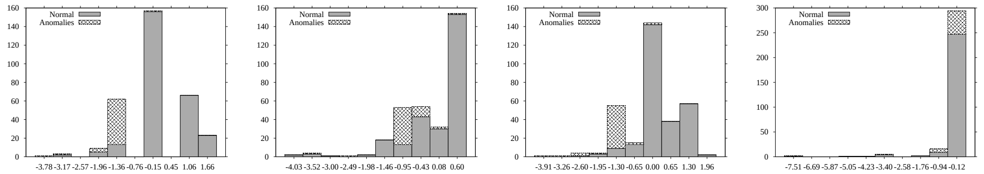
The $kNNPCA$ technique performs better on average than all 14 data driven similarity measures and the $kNNEuc$ technique. This shows that PCA is able to capture a better combination of the separability statistics automatically than captured by the similarity measures. Moreover, it also shows that using all statistics may not be optimal for several data sets, and an optimal subset is required to be selected. For some data sets we observe that $kNNPCA$ does not perform as well as using a single best discriminating statistic which is shown in the row *stt* (the corresponding statistic is shown in row *ind*). This shows that PCA might not always be able to determine the best combination of the statistics.

The performance of the best statistic (row *stt*) is the best for most of the data sets. In some cases, such as $sk1$ and $ms2$, the combination of statistics (using *Euclidean* distance or PCA) outperforms the single best statistic.

4.7.1 Designing a Better Similarity Measure

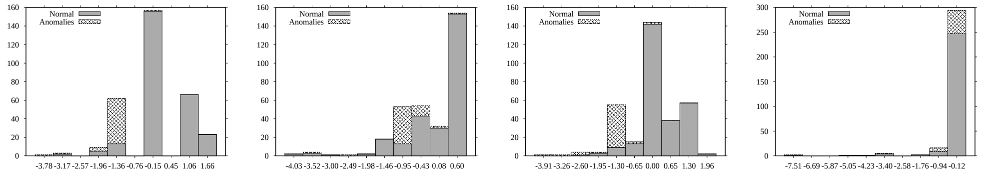
The results in Table 4.7 show that for many data sets, a combination of the separability statistics can result in better performance than using them individually. PCA is one way to obtain such a combination, but as the results indicate, it might not always provide the optimal combination. If a labeled validation data set is present, one can visually inspect the histograms for different statistics, and select the ones that provide maximum separation between the normal and anomalous data instances. We argue that using this approach we can arrive at an optimal subset of separability statistics. A similarity measure can then be designed to use this subset.

To verify the above hypothesis we conducted the following experiment. We selected data sets $sk1$ and $sk2$ from Table 4.7. For each data set, the test data is split into equal sized validation and test sets. We first map the validation set into continuous space and analyze the histograms for each separability statistic, making use of the labels for the validation instances. We then select a subset of the statistics that best separate the normal points and anomalies. Figures 4.6 and 4.7 show the per-statistic histograms for $sk1$ and $sk2$ data sets, respectively.



(a) d_{mk} statistic. (b) $f_m k$ statistic. (c) f_{xk} statistic. (d) n_{xk} statistic.

Figure 4.6: Histograms of separability statistics for data set $sk1$.



(a) d_{mk} statistic. (b) f_{mk} statistic. (c) f_{xk} statistic. (d) n_{xk} statistic.

Figure 4.7: Histograms of separability statistics for data set $sk2$.

We observe that for $sk1$, statistics 1 and 3 (d_{mk} and f_{xk}) show maximum separability between normal and anomalous data instances in the corresponding validation data set. Similarly, for $sk2$, statistics 3 and 4 (f_{xk} and n_{xk}) show maximum separability between normal and anomalous data instances in the validation data set. We then apply the *Euclidean* distance based kNN technique on the test data set using the best subset of statistics.

	$sk1$		$sk2$	
	Val.	Test	Val.	Test
ovr	0.71	0.69	0.71	0.69
$gd4$	0.16	0.14	0.12	0.16
of	0.78	0.74	0.82	0.74
esk	0.68	0.72	0.73	0.70
d_{mk}	0.71	0.69	0.71	0.69
f_{mk}	0.16	0.18	0.43	0.41
f_{xk}	0.75	0.78	0.79	0.69
n_{xk}	0.56	0.53	0.79	0.49
euc	0.73	0.61	0.84	0.78
pca	0.75	0.63	0.84	0.78
eucs	**0.84**	**0.82**	**0.84**	**0.82**

Table 4.8: Anomaly detection performance for $sk1$ and $sk2$ data sets ($k = 10$). Row *eucs* shows the results using the best subset of statistics.

Table 4.8 summarizes the performance of kNN using different similarity measures and the performance of kNN using the best subset of statistics on the two data sets. The results show that while none of the statistics individually perform as well (maximum accuracy is 0.78 for f_{xk} in $sk1$ and 0.69 for f_{xk} in $sk2$), the combination of the two best statistics (from the histograms as well as results of statistics on the validation data set), has accuracy of 0.82 for both data sets.

The results also indicate that the other two combination methods, *viz.*, *Euclidean* and PCA, are slightly worse than the optimal combination, but still outperform all similarity measures as well as the individual statistics.

Thus, given a validation data set, a better subset of statistics can be selected by either using the histograms or by observing the results of individual statistics on the validation data set.

4.8 Concluding Remarks and Future Research Directions

This chapter presents a framework to analyze categorical data. It is clear from the discussion in the previous sections that there is a tremendous gap between exploratory data analysis techniques for continuous and categorical data sets. The RBA framework is an attempt towards bridging this gap. By mapping categorical data to continuous space, we open up the possibility of utilizing exploratory techniques that are available for continuous data to be applied to categorical data. The key strength of the proposed framework is its ability to analyze a given test data set with respect to a reference data set. We have demonstrated how this property can be used for visualization as well as anomaly detection. In both applications, the framework is used to distinguish between instances belonging to the reference class(es) against the instances belonging to a novel class. Visualization allows an analyst to understand the data, set optimal parameters (such as number of nearest neighbors k), as well as choose or design optimal similarity measures using the proposed statistics. We believe that this framework can be extended in several directions, and discuss some future directions for research here.

The separability statistics, discussed in Section 4.4, are inspired from different similarity measures that have been proposed for categorical data. Many other such canonical statistics can be developed, which can be used to distinguish between instances that belong to the reference class against the other instances, e.g., a statistic that captures the correlation between different attributes.

Note that each of the separability statistics as well as their combinations can serve as distance/similarity measures. We showed that one can select an appropriate subset of separability statistics (or their linear combination, e.g. using PCA) in a supervised setting. This opens up the possibility for devising entirely new distance/similarity measures for categorical data sets.

In this chapter we have used two standard visualization techniques, viz., histograms and 2-D plots of data projected on top two principal components. Other visualization and exploratory techniques that are applied to continuous data (see Table 4.2, [114], [170]), can also be applied to the mapped data.

We have demonstrated the discriminative power of the framework in the context of anomaly detection, but one can extend it for other data mining tasks such as classification and clustering. Moreover, the concept of analyzing a test data set with respect to a reference data set can also be extended to other type of data such as multivariate continuous data or sequence data. Specifically, using a set of separability statistics (similar to the ones proposed in Section 4.4), any type of data can also be analyzed in the same framework. We will show an application of RBA in analyzing symbolic sequence data sets in Chapter 6.

Another possible extension to the proposed framework is to analyze a given data set with respect to itself. If the data mostly contains instances belonging to one or a few dominant modes, and a few anomalies, the anomalies should, in principle, appear different than the normal instances in the mapped space. Thus, the framework can be used for tasks such as unsupervised anomaly detection or noise removal.

Part II

Detecting Anomalies in Symbolic Sequences

Chapter 5

Anomaly Detection Techniques for Symbolic Sequences – A Comparative Evaluation

A large number of anomaly detection techniques for symbolic sequences have been proposed as shown in Table 5. The techniques can be classified into three broad categories based on the underlying approach. *Kernel based* techniques assign an anomaly score to a test sequence based on its similarity to the normal sequences. *Window based* techniques calculate the probability of occurrence of every fixed length window in the test sequence. *Markovian* techniques calculate the probability of occurrence of each symbol in the test sequence conditioned on the preceding few symbols in the test sequence. As Table 5 shows, these techniques have been developed in the context of different domains. For example, Sun et al [169] proposed a probabilistic suffix trees (PST) based technique to detect anomalous sequences in a data base of protein sequences. Forrest et al proposed several techniques to detect anomalous sequences in a data base of operating system call sequences [58, 87, 59]. While the techniques were proposed and evaluated in specific domains, no systematic evaluation is available regarding their relative performance. In particular, it is unclear if the techniques are the best ones for the domain they were proposed for or if another techniques (originally proposed for an entire different domain) might perform better.

The reason such an evaluation is necessary is because of the difference in the nature of anomaly detection problem in different domains. The difference can exist due to following reasons:

1. The composition of the sequences collected within each domain can be very different. In system call intrusion detection domain, the maximum alphabet size for the symbols that make up the sequences is close to 160. For protein sequences and sequences collected from aircraft flights, the alphabet sizes are close 20 and 1000, respectively. The average lengths of the sequences in different domains also varies from close to 100 (for protein sequences) to as long as 1000 (for system call sequences). Moreover, in some domains, the sequences are of relatively similar lengths, such as protein sequences, while in others, such as system call intrusion detection, the lengths of sequences can vary significantly.

2. The relation between the normal sequences varies across domains.. The normal sequences can either belong to a single mode or to multiple modes. For example, for protein sequences, the normal sequences correspond to a single protein family, and hence belong to a single mode. For network intrusion detection, the sequences correspond to different types of network activities, and hence the normal sequences belong to different modes.

3. The nature of anomalies in the anomalous sequences can be different across domains. A sequence maybe anomalous because it is comes from a different generative mechanism than the normal sequences. On the other hand, an anomalous sequence may come from the same generative mechanism as the normal sequence, but deviate from the normal for a short span or duration. For example, the anomalous sequences in a protein data set belong to a different family than the normal sequences, and hence can be thought of as being generated by a very different generative mechanism. The anomalous sequences in the intrusion detection data sets correspond to scenario when the normal operation of a system is disrupted for a short span. Thus the anomalous sequences are expected to appear like normal sequences for most of the span of the sequence, but deviate in very few locations of the sequence.

Application Domains	Kernel Based Techniques	Window Based Techniques	Markovian Techniques		
			Fixed	Variable	Sparse
Intrusion Detection		[58],[87], [59],[74]	[59],[68], [147],[116], [115],[133]		[59], [52]
Proteomics				[169]	
Flight Safety	[29]		[165]		

Table 5.1: Anomaly Detection Techniques for Symbolic Sequences.

In this chapter, we provide an experimental evaluation of a large number of anomaly detection techniques for symbolic sequences on a variety of data sets/ to explain the performance of a variety of anomaly detection techniques on different types of sequence data sets. We also propose two novel anomaly detection techniques that show consistently superior performance over existing techniques across most data sets. The analysis presented in this chapter allows relative comparison of the different anomaly detection techniques and highlights their strengths and weaknesses.

We also propose a novel artificial data generator that can be used to generate validation data sets to evaluate anomaly detection techniques for sequences. The generator allows to generate data sets with different characteristics by varying the associated parameters to study the relationship between the anomaly detection techniques and the different characteristics of the data.

The rest of this chapter is organized as follows. Section 5.1 describes the different techniques that are evaluated in this chapter. Section 5.2 describes the various data sets that are used for evaluation. Section 5.3 contains the experimental results. Section 5.4 contains conclusions from the experimental findings.

5.1 Anomaly Detection Techniques for Sequences

All techniques discussed here solve the semi-supervised problem as discussed in Chapter 3. Each sequence is defined using a finite alphabet, Σ. We evaluated a variety of techniques that can be grouped into following three categories:

5.1.1 Kernel Based Techniques

Kernel based techniques make use of kernel K by using the pairwise similarity between sequences. In the problem formulation stated in Definition 1 the sequences can be of different lengths, hence simple measures such as *Hamming Distance* cannot be used. One possible measure is the normalized length of *longest common subsequence* between a pair of sequences. This similarity between two sequences S_1 and S_2, is computed as:

$$nLCS(S_1, S_2) = \frac{|LCS(S_1, S_2)|}{\sqrt{|S_1||S_2|}} \tag{5.1}$$

Since the value computed above is between 0 and 1, the distance between S_1 and S_2 can be computed as [172]:

$$d(S_1, S_2) = 1 - nLCS(S_1, S_2) \tag{5.2}$$

Other similarity measures other than $nLCS$ can be used as well, e.g., time series bitmaps [109], *string kernels* [], such as the *spectrum kernel* [118]. We use $nLCS$ in our experimental study, since it was used in [29] to detect anomalies in discrete sequences and appears promising.

Computing Kernel For a given test data set, a kernel matrix $K \in \Re^{m \times n}$ is computed such that:

$$K[i][j] = nLCS(S_i, S_j),\ S_i \in \mathbf{S},\ S_j \in \mathbf{T} \tag{5.3}$$

Nearest Neighbors Based (kNN)

In the nearest neighbor scheme (kNN), for each test sequence $S_i \in \mathbf{S}$, the distance to its k^{th} nearest neighbor in the training set $\mathbf{T}$ is computed using the kernel matrix K. This distance becomes the anomaly score $A(S_i)$ [172, 149].

Clustering Based ($CLUSTER$)

This technique clusters the sequences in $\mathbf{T}$ into a fixed number of clusters, c, by using the kernel matrix K. The test phase involves measuring the distance of every test sequence, $S_i \in \mathbf{S}$, with the medoid of each cluster. The distance to the medoid of the closest cluster becomes the anomaly score $A(S_i)$.

5.1.2 Window Based Techniques

Window based techniques try to localize the cause of anomaly in a test sequence, within one or more windows, where a window is a fixed length subsequence of the test sequence. One such technique called *Threshold Sequence Time-Delay Embedding* (*tSTIDE*) [59] uses a sliding window of fixed size k to extract k-length windows from the training sequences in $\mathbf{T}$. The count of each window occurring in $\mathbf{T}$ is maintained. During testing, k-length windows are extracted from a test sequence S_i. Each such window ω_i is assigned a likelihood score $P(\omega_i) = \frac{f(\omega_i)}{f(*)}$, where $f(\omega_i)$ is the frequency of occurrence of window ω_i in $\mathbf{T}$, and $f(*)$ is the total number of k length windows extracted from $\mathbf{T}$.

For the test sequence S_i, $|S_i| - k + 1$ windows are extracted, and a likelihood score vector of length $|S_i| - k + 1$ is obtained. This score vector is then combined to obtain the anomaly score for the sequence, $A(S_i)$, in the following way:

$$L(S_i) = \frac{1}{|S_i| - k + 1} \sum_{i=1}^{|S_i|-k+1} \log P(\omega_i) \tag{5.4}$$

$$A(S_i) = -1 * L(S_i) \tag{5.5}$$

If likelihood score for any window is 0, it is replaced with 10^{-6} since $\log 0$ is undefined. Other alternatives to combine the score vector to obtain $A(S_i)$ are discussed in Section 5.1.4.

5.1.3 Markovian Techniques

Such techniques estimate the conditional probability for each symbol in a test sequence S_i conditioned on the symbols preceding it. Most of the techniques utilize the *short memory* property of sequences [151]. This property is a higher-order Markov condition which states that for a given sequence $S = \langle s_1, s_2, \ldots s_{|S|} \rangle$, the conditional probability of occurrence of a symbol s_i is given as:

$$P(s_i|s_1 s_2 \ldots s_{i-1}) = P(s_i|s_{i-k+1} \ldots s_{i-1}), i > k \tag{5.6}$$

In the following, we investigate four Markovian techniques. Each one of them computes a vector of scores, each element of which corresponds to the conditional probability of observing a symbol, as defined in (5.6). This score vector is then combined to obtain $A(S_i)$ using equations similar to (5.4) and (5.5), by replacing $P(\omega_i)$ with $P(s_i|s_{i-k+1}\ldots s_{i-1})$.

Fixed Length Markovian Technique

A fixed length Markovian technique determines the probability $P(s_{qi})$ of a symbol s_{qi}, conditioned on a fixed number of preceding symbols[1]. One such technique uses an extended Finite State Automaton (FSA) to estimate the conditional probabilities. We will refer to this technique as FSA in subsequent discussions.

FSA extracts $(n+1)$ sized subsequences from the training data $\mathbf{T}$ using a sliding window. Each node in the automaton constructed by FSA corresponds to a unique subsequence of n symbols that form the first n symbols of such $(n+1)$ length subsequences. An edge exists between a pair of nodes, N_i and N_j in the FSA, if N_i corresponds to states $s_{i1}s_{i2}\ldots s_{in}$ and N_j corresponds to states $s_{i2}s_{i3}\ldots s_{in}s_{jn}$. At every state of the FSA two quantities are maintained. One is the number of times the n length subsequence corresponding to the state is observed in $\mathbf{T}$. The second quantity is a vector of frequencies corresponding to number of times different edges emanating from this state are observed. Using these two quantities, the conditional probability for a symbol, given preceding n symbols, can be determined.

During testing, the automaton is used to determine a likelihood score for every $n+1$ subsequence extracted from test sequence S_i which is equal to the conditional probability associated with the transition from the state corresponding to first n symbols to the state corresponding to the last n symbols. If there is no state in the automaton corresponding to the first n symbols, the subsequence is ignored.

$FSAz$ We propose a variant of FSA technique, in which if there is no state, learnt from the training set, corresponding to the first n symbols of a $n+1$ subsequence, we assign a low score (e.g. 0) to that subsequence, instead of ignoring it. The intuition behind assigning a low score to non-existent states is that anomalous test sequences are more likely to contain such states, than normal test sequences. While FSA ignores this information, we utilize it in $FSAz$.

---

[1] A more general formulation that determines probability of l symbols conditioned on a fixed number of preceding n symbols is discussed in [133].

Probabilistic Suffix Trees (PST)

A PST is a compact tree representation of a variable Markov chain, which uses classical *suffix trees* as its index structure [151]. We evaluate a PST based anomaly detection technique [169], that learns a PST from the training sequences and then assigns a conditional likelihood score to each symbol of the test sequence.

In a PST, each edge is labeled using a symbol, and each node represents the subsequence obtained by traversing the path from root to the node, as well as the number of times the subsequence is observed in the training sequences. Each node also stores the conditional probability of observing each symbol in the alphabet, given the subsequence represented by the node. The PST is grown (training phase) by scanning the training sequences. The maximum depth of the tree is fixed at k, which is a user defined parameter. Several pruning criterion are applied to the PST to ensure that the PST contains only those paths that occur significantly enough number of times in the training sequences. The pruning can be done by applying thresholds to the frequency of a node label, or to the conditional probability of a symbol emanating from a given node. If no pruning is applied, the PST is equivalent to the $FSAz$.

During the testing phase for the PST based technique the test sequence is scanned and the PST is traversed simultaneously. For a symbol s_{qi} in the test sequence S_i, its conditional probability is estimated by finding the longest suffix of the k length subsequence that precedes s_{qi} (in S_i) and occurs as a path in the PST. Thus, different symbols are conditioned on a different sized history.

Sparse Markovian Technique

Sparse Markovian techniques are more flexible than variable Markovian techniques, in the sense that they estimate the conditional probability of s_{qi} based on a subset of symbols within the preceding k symbols, which are not necessarily contagious to s_{qi}. In other words the symbols are conditioned on a sparse history.

[116] use $RIPPER$ classifier to build one such sparse model. In this approach, a sliding window is applied to the training data $\mathbf{T}$ to obtain k length windows. The first $k-1$ positions of these windows are treated as $k-1$ categorical attributes, and the k^{th} position is treated as a target class. $RIPPER$ [44] is used to learn rules that can

predict the k^{th} symbol given the first $k-1$ symbols. To ensure that there is no symbol that occurs very rarely as the target class, the training sequences are duplicated 5 times.

For testing, k length subsequences are extracted from each test sequence S_i using a sliding window. For any subsequence, the first $k-1$ events are classified using the classifier learnt in the training phase and the prediction is compared to the k^{th} symbol. $RIPPER$ also assigns a confidence score associated with the classification, denoted as $conf(s_{qi}) = \frac{100T}{M}$, where M is the number of times the particular rule was fired in the training data, and T is the number of times the rule gave correct prediction. [116] assign the likelihood score of symbol s_{qi} as follows:

- For a correct classification, $P(s_{qi}) = 1$.
- For a misclassification, $P(s_{qi}) = \frac{1}{conf(s_{qi})} = \frac{M}{100T}$.

Hidden Markov Models Based Technique (HMM)

Techniques that apply HMMs for modeling sequences, transform an input sequence from the symbol space to the hidden state space. The key assumption for the HMM based anomaly detection technique [59] is that the normal sequences can be effectively represented in the hidden state space, while anomalous sequences cannot be.

The training phase involves learning an HMM with σ hidden states, from the normal sequences in $\mathbf{T}$ using the *Baum Welch* algorithm.In the testing phase, the optimal hidden state sequence for the given input test sequence S_i is determined, using the *Viterbi* algorithm.For every pair of consecutive states, $\langle s^H_{qi}, s^H_{qi+1}\rangle$, in the optimal hidden state sequence, the state transition matrix provides a likelihood score for transitioning from s^H_{qi} to s^H_{qi+1}. Thus a likelihood score vector of length $|S_i| - 1$ is obtained.

5.1.4 Combining Scores

For each of the window based and Markovian techniques, a likelihood score vector is generated for a test sequence, S_i. A combination function is then applied to obtain a single anomaly score $A(S_i)$. In Section 5.1.2, we presented one such combination technique, average log score, which was originally used in the PST technique [169]. $L(S_i)$ can be computed in other ways, such as average score [115], minimum score, maximum score, and using a threshold [133, 59]. For the threshold method, a user defined threshold is employed to determine which scores in the likelihood score vector are anomalous. The number of such anomalous scores is the anomaly score $A(S_i)$ of the test sequence. Setting the threshold often requires experimenting with different possible values, and then choosing the best performing value.

5.2 Data Sets Used

In this section we describe various public as well as the artificially generated data sets that we used to evaluate the different anomaly detection techniques. We used public data sets that have been used earlier to evaluate sequence anomaly detection techniques. To further illustrate certain aspects of different techniques, we constructed different artificial data sets. The artificial data sets were constructed such that we can control the nature of normal as well as anomalous sequences and hence learn the relationship between the various techniques and the nature of the data.

For every data set, we first constructed a set of normal sequences, and a set of anomalous sequences. A sample of the normal sequences was used as training data for different techniques. A disjoint sample of normal sequences and a sample of anomalous sequences were added together to form the test data. The relative proportion of normal and anomalous sequences in the test data determined the "difficulty level" for that data set. We experimented with different ratios such as 1:1, 10:1 and 20:1 of normal and anomalous sequences. Results on data sets with other ratios are consistent in relative terms, although most techniques perform much better for the simplest data set that uses a ratio 1:1. Since in real sequences anomalies are rare, we report results when normal and anomalous sequences were in 20:1 ratio in test data. In reality, the ratio of normal to anomalous can be even larger than 20:1. But we were unable to try such skewed distributions due to limited number of normal samples available in some of the data sets.

5.2.1 Public Data Sets

Table 5.2 summarizes the various statistics of the data sets used in our experiments. All data sets are available from our web site[2]. The distribution of the symbols for normal

---

[2] `http://www.cs.umn.edu/~chandola/ICDM2008`

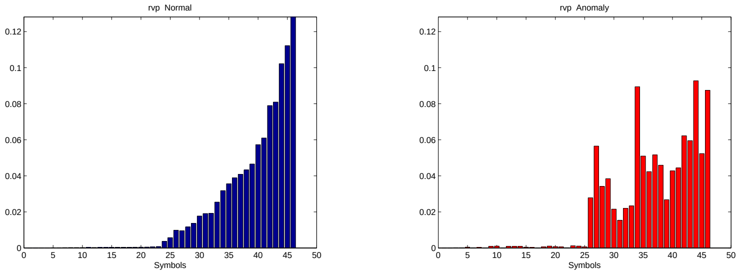
Source	Data Set	$\|\Sigma\|$	$\hat{l}$	$\|\mathbf{S^N}\|$	$\|\mathbf{S^A}\|$	$\|\mathbf{T}\|$	$\|\mathbf{S}\|$
PFAM	HCV	44	87	2423	50	1423	1050
	NAD	42	160	2685	50	1685	1050
	TET	42	52	1952	50	952	1050
	RUB	42	182	1059	50	559	525
	RVP	46	95	1935	50	935	1050
UNM	snd-cert	56	803	1811	172	811	1050
	snd-unm	53	839	2030	130	1030	1050
DARPA	bsm-week1	67	149	1000	800	10	210
	bsm-week2	73	141	2000	1000	113	1050
	bsm-week3	78	143	2000	1000	67	1050

Table 5.2: Public data sets used for experimental evaluation. $\hat{l}$ – Average Length of Sequences, $\mathbf{S^N}$ – Normal Data, $\mathbf{S^A}$ – Anomalous Data, $\mathbf{T}$ – Training Data, $\mathbf{S}$ – Test Data.

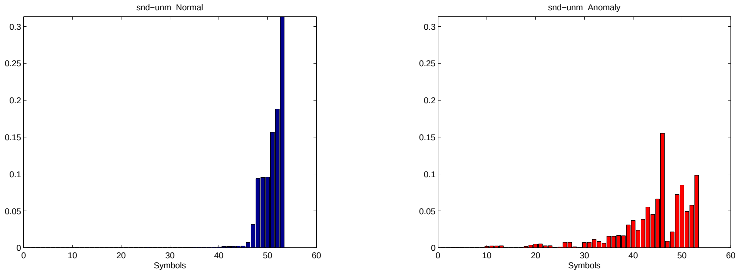
and anomalous sequences is illustrated in Figures 5.1(a),5.1(b) (RVP), 5.1(c),5.1(d) (snd-unm), and 5.1(e),5.1(f), (bsm-week2). The distribution of symbols in snd-unm data is different for normal and anomalous data, while the difference is not significant in RVP and bsm-week2 data. We will explain how the normal and anomalous sequences were obtained for each type of data set in the next subsections.

Protein Data Sets

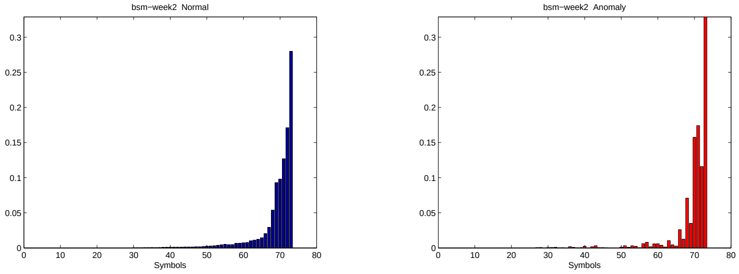
The first set of public data sets were obtained from PFAM database (Release 17.0) [15] containing sequences belonging to 7868 protein families. Sequences belonging to one family are structurally different from sequences belonging to another family. We choose five families, viz., HCV, NAD, TET, RVP, RUB. For each family we construct a normal data set by choosing a sample from the set of sequences belonging to that family. We then sample 50 sequences from other four families to construct an anomaly data set. Similar data was used by [169] to evaluate the *PST* technique. The difference was that the authors constructed a test data for each pair of protein families such that samples from one family were used as normal and samples from the other were used as test. The *PST* results on PFAM data sets reported in this chapter appear to be worse than those reported in [169].



(a) RVP Normal (b) RVP Anomalous



(c) snd-cert Normal (d) snd-cert Anomalous



(e) bsm-week2 Normal (f) bsm-week2 Anomalous

Figure 5.1: Distribution of Symbols in Training Data Sets of Different Types.

Intrusion Detection Data Sets

The second set of public data sets were collected from two repositories of benchmark data generated for evaluation of intrusion detection algorithms. One repository was generated at University of New Mexico[3]. The normal sequences consisted of sequence of system calls generated in an operating system during the normal operation of a computer program, such as sendmail, ftp, lpr etc. The anomalous sequences consisted of sequence of system calls generated when the program is run in an abnormal mode, corresponding to the operation of a hacked computer. We report results on two data sets, viz, *snd-unm* and *snd-cert*. Other data sets were not used due to insufficient anomalous sequences to attain a 20:1 imbalance. For each of the two data sets, the number of sequences in the normal as well as anomaly data was small (less than 200), making it difficult to construct significant test and training data sets. The increase the size of the data sets, we extracted subsequences of length 100 by sliding a window of length 100 and a sliding step of 50. The subsequences extracted from the original normal sequences were treated as normal sequences and the subsequences extracted from the original anomalous sequences were treated as anomalous sequences if they did not occur in the normal sequences.

The other intrusion detection data repository was the *Basic Security Module* (BSM) audit data, collected from a victim Solaris machine, in the DARPA Lincoln Labs 1998 network simulation data sets [121]. The repository contains labeled training and testing DARPA data for multiple weeks collected on a single machine. For each week we constructed the normal data set using the sequences labeled as normal from all days of the week. The anomaly data set was constructed in a similar fashion. The data is similar to the system call data described above with similar (though larger) alphabet.

5.2.2 Altered RVP Data Set

To better understand the performance of the anomaly detection techniques to the nature of anomalies in the test data, we created a data set from the original RVP data from the PFAM repository. A test data set was constructed by sampling 800 most normal sequences not present in training data. Anomalies were injected in 50 of the test

---

[3] `http://www.cs.unm.edu/~immsec/systemcalls.htm`

sequences by randomly replacing k symbols in each sequence with the least frequent symbol in the data set. The parameter k controls the deviation of the anomalous sequences from the normal sequences. The objective of this experiment was to evaluate how the performance of a technique varies with k.

5.2.3 Artificial Data Sets

As mentioned in the introduction, two types of anomalous sequences can exist, one which are arguably generated from a different generative mechanism than the normal sequences, and the other which result from a normal sequence deviating for a short span from its expected normal behavior. To study the relationship between these two types of anomalous sequences and the performance of different techniques, we designed an artificial data generator which allows us to generate validation data sets with different types of anomalies.

We used a generic HMM, as shown in Figure 5.2 to model normal as well as anomalous data. The HMM shown in Figure 5.2 has two sets of states, $\{S_1, S_2, \dots S_6\}$ and $\{S_7, S_8, \dots S_{12}\}$.

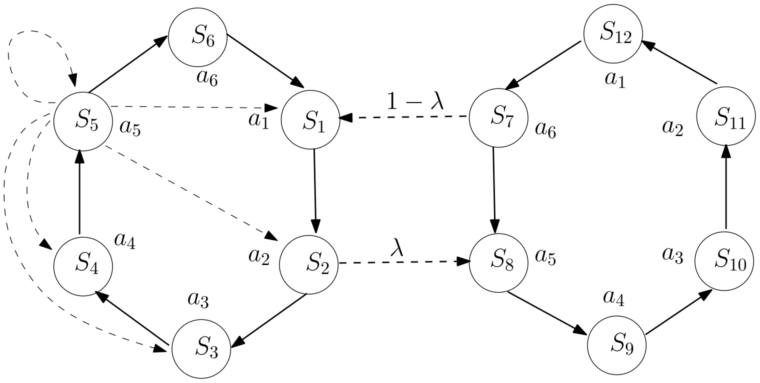


Figure 5.2: HMM used to generate artificial data.

Within each set, the transitions corresponding to the solid arrows shown in Figure 5.2 are assigned a transition probability of $(1-5\beta)$, while other transitions are assigned transition probability β. These transitions are shown as dotted line for only one state, S_5. No transition is possible between states belonging to different sets. The only exception are S_2S_8 for which the transition probability is λ, and S_7S_1 for which the transition probability is $1 - \lambda$. The transition probabilities S_2S_3 and S_7S_8 are adjusted accordingly so that the sum of transition probabilities for each state is 1.

The observation alphabet is of size 6. Each state emits one alphabet with a high probability $(1 - 5\alpha)$, and all other alphabets with a low probability (α). Figure 5.2 depicts the most likely alphabet for each state.

The initial probability vector π of the HMM is constructed such that either $\pi_1 = \pi_2 = \ldots = \pi_6 = 1$ and $\pi_7 = \pi_8 = \ldots = \pi_{12} = 0$; or vice-versa.

After manually constructing the HMM, as described above, it is used to generate random sequences of desired lengths. Normal sequences are generated by setting λ to a low value and π to be such that the first 6 states have initial probability set to $\frac{1}{6}$ and rest 0. If $\lambda = \beta = \alpha = 0$, the normal sequences will consist of the subsequence $a_1a_2a_3a_4a_5a_6$ getting repeated multiple times. By increasing λ or β or α, anomalies can be induced in the normal sequences.

This generic HMM can be tuned to generate two types of anomalous sequences. For the first type of anomalous sequences, λ is set to a high value and π to be such that the last 6 states have initial probability set to $\frac{1}{6}$ and rest 0. The resulting HMM is directly opposite to the HMM constructed for generating normal sequences. Hence the anomalous sequences generated by this HMM are completely different from the normal sequences. Such anomalous sequences are motivated from the anomalies found in protein data sets.

To generate second type of anomalous sequences, the HMM used to generate the normal sequence is used, with the only difference that λ is increased to a higher value than 0. Thus the anomalous sequences generated by this HMM will be similar to the normal sequences except that there will be short spans when the symbols are generated by the second set of states. Such anomalous sequences are motivated from the anomalies found in system call intrusion detection data sets.

By varying λ, β, and α, we generated several evaluation data sets (with two different type of anomalous sequences). We will present the results of our experiments on these artificial data sets in next section.

5.3 Experimental Results

The experiments were conducted on a variety of data sets discussed in Section 5.2. The various parameter settings associated with each technique were explored. The results presented here are for the parameter setting which gave best results across all data sets, for each technique.

5.3.1 Sensitivity to Parameters

The performance of $CLUSTER$ improved as c was increased from 2 onwards, but stabilized for values greater than 32. The best overall performance was observed for $c = 32$. For kNN, the performance was comparable for a wide range of k ($2 \leq k \leq 32$) but deteriorated for higher values of k. The best overall performance was observed for $k = 4$. For $tSTIDE$ as well as the Markovian techniques (FSA, $FSAz$, PST, $RIPPER$), the performance was sensitive to the choice of window length or the length of the history. For low values of this length (≤ 5) or for values higher than 10, the performance was generally poor. The best performing setting was window size of 6 for $tSTIDE$ and history length of 5 for the Markovian techniques. For PST, an additional parameter is P_{min} which controls the threshold under which the counts for a given subsequence are considered insignificant. We observed that performance of PST was highly sensitive to this parameter. If P_{min} was set to very low (≈ 0), PST performed similar to $FSAz$, while if P_{min} was set to be higher than 0.1, the performance was poor. The best performance of PST was observed for $P_{min} = 0.01$. For HMM, the number of hidden states σ is a critical parameter. We experimented with values ranging from 2 to $|\Sigma|$. Our experiments reveal that the performance of HMM does not vary significantly for different values of σ. The best overall performance of HMM was observed for $\sigma = 4$ for public data sets and $\sigma = 12$ for the artificial data sets.

We experimented with various combination functions for different techniques, and found that the *average log score* function has the best performance across all data sets. Hence, results are reported for the *average log score* function. Results with other combination techniques are available in our technical report [39].

5.3.2 Accuracy vs. AUC

We evaluated the different techniques using the two evaluation metrics described in Chapter 3, *Accuracy* and *AUC*. Both metrics show similar relative performance for the different techniques. We will compare the performance using the *accuracy* metric.

5.3.3 Results on Public Data Sets

Tables 5.3 and 5.4 summarize the accuracy and AUC results on the 10 public data sets. $CLUSTER$ and kNN show good performance for PFAM and UNM data sets but perform moderately on DARPA data sets. FSA and $FSAz$ show consistently good performance for all public data sets. $tSTIDE$ performs well for PFAM data sets but its performance degrades for both UNM and DARPA data sets. PST performs average to poor for all data sets including the PFAM data sets for which it was originally used. The HMM technique performs poorly for all public data sets. The reasons for the poor performance is that HMM technique makes an assumption that the normal sequences can be represented with σ hidden states, which might not be true for the public data sets.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
cls	0.54	0.46	0.84	0.86	0.76	0.76	**0.94**	0.20	0.36	0.52	0.62
knn	0.88	0.64	**0.86**	**0.90**	0.72	**0.84**	**0.94**	0.20	0.52	0.48	0.70
tstd	0.90	**0.74**	0.50	**0.90**	**0.88**	0.58	0.64	0.20	0.36	0.60	0.63
fsa	0.88	0.66	0.48	**0.90**	0.80	0.82	0.88	0.40	0.52	0.64	0.70
fsaz	**0.92**	0.72	0.50	**0.90**	**0.88**	0.80	0.88	**0.50**	**0.56**	**0.66**	**0.73**
pst	0.74	0.10	0.66	0.50	0.28	0.28	0.10	0.00	0.10	0.34	0.31
rip	0.52	0.20	0.36	0.66	0.72	0.72	0.70	0.20	0.18	0.50	0.48
hmm	0.10	0.06	0.20	0.10	0.00	0.00	0.00	0.00	0.02	0.20	0.07
Avg	0.69	0.45	0.55	0.72	0.63	0.60	0.64	0.21	0.33	0.49	

Table 5.3: Accuracy results for public data sets.

Overall, one can observe that the performance of techniques in general is better for PFAM data sets and on UNM data sets, while the DARPA data sets are more challenging.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
cls	0.98	0.96	1.00	1.00	0.99	0.99	1.00	0.74	0.90	0.91	0.94
knn	1.00	0.98	1.00	1.00	0.99	1.00	1.00	0.75	0.92	0.91	0.95
tstd	0.99	0.97	0.98	1.00	1.00	0.97	0.92	0.62	0.73	0.80	0.90
fsa	0.98	0.97	0.92	0.99	0.99	0.99	0.96	0.88	0.90	0.97	0.96
fsaz	1.00	0.98	0.98	1.00	1.00	0.97	0.96	0.88	0.91	0.97	0.96
pst	0.99	0.54	0.98	0.97	0.91	0.93	0.88	0.35	0.42	0.54	0.75
rip	0.70	0.45	0.37	0.97	0.96	0.98	0.94	0.79	0.70	0.84	0.77
hmm	0.58	0.50	0.71	0.55	0.24	0.04	0.03	0.43	0.50	0.77	0.43
Avg	0.90	0.79	0.87	0.93	0.88	0.86	0.84	0.68	0.75	0.84	

Table 5.4: AUC results for public data sets.

5.3.4 Results on Altered RVP Data Set

Figure 5.3 shows the performance of the different techniques on the altered RVP data set, for different values of k from 1 to 10. We observe that $FSAz$ performs remarkably well for these values of k. $CLUSTER$, $tSTIDE$, FSA, PST, and $RIPPER$ exhibit moderate performance, though for values of k closer to 10, $RIPPER$ performs better than the other 4 techniques. For $k > 10$, all techniques show better than 90% accuracy because the anomalous sequences become very distinct from the normal sequences, and hence all techniques perform comparably well.

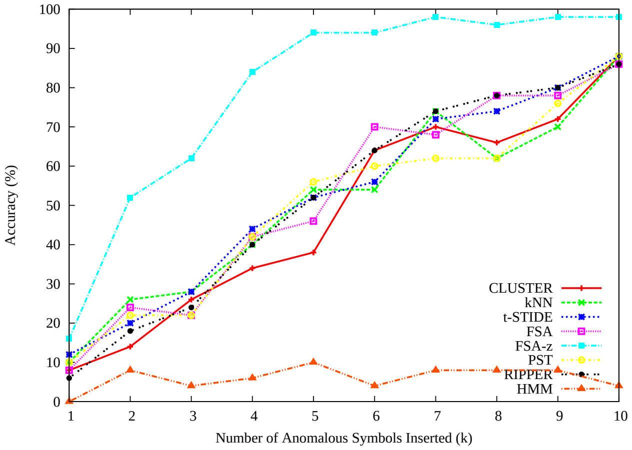


Figure 5.3: Results for altered RVP data sets

5.3.5 Results on Artificial Data Sets

Tables 5.5 and 5.6 summarize the accuracy and AUC results on 6 ($d1$–$d6$) artificial data sets.The normal sequences in data set $d1$ were generated with $\lambda = 0.01, \beta = 0.01, \alpha = 0.01$. The anomalous sequences were generated using the first setting as discussed in Section 5.2.3, such that the sequences were primarily generated from the second set of states. For data sets $d2$–$d6$, the HMM used to generate normal sequences was tuned with $\beta = 0.01, \alpha = 0.01$. The value of λ was increased from 0.002 to 0.01 in increments of 0.002. The anomalous sequences for data sets $d2$–$d6$ were generated using the second setting in which λ is set to 0.1.

	d1	d2	d3	d4	d5	d6	**Avg**
cls	**1.00**	0.80	0.74	0.74	0.58	0.64	0.75
knn	**1.00**	0.88	0.76	**0.76**	0.60	**0.68**	0.78
tstd	**1.00**	0.82	0.64	0.64	0.48	0.50	0.68
fsa	**1.00**	0.88	0.50	0.52	0.24	0.28	0.57
fsaz	**1.00**	**0.92**	0.60	0.52	0.32	0.38	0.62
pst	**1.00**	0.84	**0.82**	**0.76**	**0.68**	**0.68**	**0.80**
rip	**1.00**	0.78	0.64	0.66	0.52	0.44	0.67
hmm	**1.00**	0.50	0.34	0.42	0.16	0.66	0.51
Avg	1.00	0.80	0.63	0.63	0.45	0.53	

Table 5.5: Accuracy results for artificial data sets.

	d1	d2	d3	d4	d5	d6	**Avg**
cls	1.00	0.95	0.97	0.98	0.94	0.95	0.97
knn	1.00	0.96	0.98	0.98	0.96	0.95	0.97
tstd	1.00	0.96	0.98	0.98	0.96	0.95	0.97
fsa	1.00	0.96	0.98	0.98	0.96	0.95	0.97
fsaz	1.00	0.96	0.98	0.98	0.96	0.95	0.97
pst	1.00	0.96	0.98	0.98	0.96	0.95	0.97
rip	1.00	0.96	0.98	0.98	0.96	0.95	0.97
hmm	1.00	0.96	0.98	0.98	0.96	0.95	0.97
Avg	1.00	0.96	0.98	0.98	0.96	0.95	

Table 5.6: AUC results for artificial data sets.

From Table 5.5, we observe that PST is the most stable technique across the artificial data sets, while the deterioration is most pronounced for FSA and $FSAz$. Both kNN and $CLUSTER$ also get negatively impacted as the λ increases but the trend is gradual than for $FSAz$. The performance of HMM on the artificial data sets is better than for public data sets since the training data was actually generated by a 12 state HMM

and the HMM technique was trained with $\sigma = 12$; thus the HMM model effectively captures the normal sequences.

5.3.6 Relative Performance of Different Techniques

Kernel based techniques are found to perform well for data sets in which the anomalous sequences are significantly different from the normal sequences; but perform poorly when the different between the two is small. This is due to the nature of the normalized LCS similarity measure used in the kernel based techniques. Our experiments show that kNN technique is somewhat better suited than $CLUSTER$ for anomaly detection, which is expected, since kNN is optimized to detect anomalies while the clustering algorithm in $CLUSTER$ is optimized to obtain clusters in the data.

$FSAz$ is consistently superior among all techniques, especially for data sets in which the anomalous sequences are minor deviations from normal sequences. The performance of $FSAz$ is poor when the normal sequences contain rare patterns. $FSAz$ is consistently superior to FSA. Performance of $tSTIDE$ is comparable to $FSAz$ when the anomalous sequences are significantly different from the normal sequences, but is inferior to $FSAz$ when the difference is small. $tSTIDE$ is less affected by the presence of rare patterns in the normal sequences than $FSAz$. for all PFAM data sets but is relatively poor on DARPA and UNM data sets. $tSTIDE$ performs significantly better on artificial data sets. PST performs relatively worse than other techniques, except for cases where the normal sequences themselves contain many rare patterns. $RIPPER$ is also an average performer on most of the data sets, and is relatively better than PST, indicating that using a sparse history model is better than a variable history model.

For the public data sets, we found the HMM technique to perform poorly. The reasons for the poor performance of HMM are twofold. The first reason is that HMM technique makes an assumption that the normal sequences can be represented with σ hidden states. Often, this assumption does not hold true, and hence the HMM model learnt from the training sequences cannot emit the normal sequences with high confidence. Thus all test sequences (normal and anomalous) are assigned a low probability score. The second reason for the poor performance is the manner in which a score is assigned to a test sequence. The test sequence is first converted to a hidden state sequence, and then a $1+1$ FSA is applied to the transformed sequence. We have observed from our experiment using FSA that a $1+1$ FSA does not perform well for anomaly detection. The performance of HMM on artificial data sets (See Table 5.5) illustrates this argument. Since the training data was actually generated by a 12 state HMM and the HMM technique was trained with $\sigma = 12$; thus the HMM model effectively captures the normal sequences. The results of HMM for artificial data sets are therefore better than for public data sets, but still slightly worse than other techniques because of the poor performance of the $1+1$ FSA. When the normal sequences were generated using an HMM, the performance improves significantly. The hidden state sequences, obtained as a intermediate transformation of data, can actually be used as input data to any other technique discussed here. The performance of such an approach will be investigated as a future direction of research.

5.4 Conclusions and Future Work

Our experimental evaluation provided us with valuable insights into strengths and weaknesses of different anomaly detection techniques. None of the techniques was found to be consistently superior to all other techniques, indicating that the performance of a technique depends on the nature of the sequence data set. The use of artificial data generator allowed us to arrive at conclusions that were not evident from the results on public data sets.

A significant result of this study is that several techniques have been shown to be quite effective in application domains for which they were not originally intended for. Techniques such as $tSTIDE$ and FSA, which were originally evaluated on system call intrusion detection data, show promising results on protein data sets. Interestingly, $tSTIDE$ performs relatively poorly on system call intrusion detection data sets.

Results on the public data sets (Table 5.3) reveal that FSA-z and FSA, are the most consistent techniques while PST and $RIPPER$ generally perform poorly. But the results on artificial data sets (Table 5.5) identify scenarios where the latter two techniques might be better suited than the former two.

Kernel based techniques are found to perform well for data sets in which the anomalous sequences are relatively different from the normal sequences; but perform poorly when the different between the two is small. This is due to the nature of the normalized LCS similarity measure used in the kernel based techniques. Future work should investigate other similarity measures that are able to capture the difference between sequences that are minor deviations of each other. Our experiments show that kNN technique is somewhat better suited than $CLUSTER$ for anomaly detection.

Consistent with the observations of other researchers [59], we found the HMM technique to perform poorly. When the normal sequences were generated using an HMM, the performance improves significantly. The hidden state sequences, obtained as a intermediate transformation of data, can actually be used as input data to any other technique discussed here. The performance of such an approach needs to be investigated and is suggested as a future direction of research.

Chapter 6

Reference Based Analysis Framework for Symbolic Sequences

The results on different data sets in Chapter 5 reveal that no one technique is clearly superior to others. Most techniques show consistency in performance on public data sets belonging to one domain but show different performance for data sets from a different domain. This indicates a relationship between the techniques and the nature of the data. In the artificial data sets generated from the data generator as well as the altered RVP data sets, we further studied this relationship by modifying the nature of the data using one or more tunable parameters. These observations motivate a deeper study of the relationship between a technique and a data set.

In this chapter, we study the relationship between the anomaly detection techniques and the nature of data. Using the RBA framework, introduced in Chapter 4, we characterize symbolic sequence data. We visualize the symbolic sequences using these characteristics which is useful to understand various aspects of the sequence data such as how different are the normal sequences from the anomalous sequences and how similar are the normal sequences to each other. We then show how different anomaly detection techniques evaluated in Chapter 5 rely on one or more of such characteristics to detect anomalies. Using these characteristics, we propose two novel anomaly detection techniques for symbolic sequences, called WIN_{1D} and WIN_{2D}, which show consistently superior performance over the existing techniques across the different data sets.

The rest of this chapter is organized as follows. In Section 6.1, we show how the RBA framework can be used to analyze symbolic sequences. Specifically, we show how the RBA framework can be used understand the relationship between different anomaly detection techniques and the nature of sequence data in Sections 6.2 and 6.3. In Section 6.4 we present two novel RBA based anomaly detection techniques for symbolic sequences.

6.1 Characterizing Sequence Data Using RBA

The RBA framework is highly applicable to the case of semi-supervised anomaly detection, where the normal class is the reference class, and the anomalous instances are the data instances that do not belong to the reference (or normal) class.

The key step in the RBA framework is to identify a transformation of a given data instance into a set of separability statistics using a reference data set. We describe different transformations here that can be used within the RBA framework to analyze symbolic sequences and how the transformations can be used to characterize a given test sequence data set in the following subsections.

6.1.1 1-D Frequency Profiles

The first transformation is motivated from window based techniques (See Chapter 5) that rely on the frequency of a k length window in a given sequence for anomaly detection. In this section we refer to a k length window as a k-window for brevity. Each k-window is associated with a frequency (denoted as f_k), i.e., the number of times it occurs in the training sequences.

A *1-D frequency profile* for a test sequence can be constructed as follows. First, all k-windows from the test sequence are extracted and their frequencies f_k are computed from the training sequences. The frequencies are “binned” into a fixed number (p) of *bins*. Since windows with $f_k = 0$ are of special interest, the first bin stores the windows with exactly $f_k = 0$. The other $p - 1$ bins divide the range between 1 and max into equal width intervals, where max is the maximum frequency of any window in the given data set. The values in each bin are normalized to lie between 0 and 1 by dividing them by the total number of windows in the given sequence. Thus each test sequence can be mapped into a $\Re^p$ space.

Characterizing a Sequence Data Set Using Average 1-D Frequency Profiles

To characterize a given test data set, we aggregate the 1-D frequency profiles. We construct the *average 1-D frequency profiles* for the normal test sequences and anomalous test sequences separately. It should be noted that the average profile might not be the best representation of the profiles. For example, let the test set contain 4 anomalous sequences. Using four bins ($p = 4$), let the frequency profiles for the four anomalous sequences be $(1.00, 0, 0, 0)$, $(0, 1.00, 0, 0)$, $(0, 0, 1.00, 0)$, and $(0, 0, 0, 1.00)$. The average frequency profile for the anomalous sequences will be $(0.25, 0.25, 0.25, 0.25)$ which does not provide an accurate representation of the actual profiles. But if the individual frequency profiles are similar to each other, the average profile will be representative.

A test sequence data set can be characterized with respect to a normal data set by taking the difference between the average 1-D frequency profiles for normal and anomalous test sequences. We will describe how this characteristic can be used to explain the behavior of window based techniques in Section 6.2.

6.1.2 2-D Frequency Profiles

The second transformation is motivated from Markovian techniques that rely on the frequency of a k length window as well as the frequency of the $k-1$ length prefix of the window, in a given sequence for anomaly detection. Thus, each k-window is associated with a tuple (f_k, f_{k-1}), where f_k is the frequency of occurrence of the k-window and f_{k-1} is the frequency of occurrence of the $k-1$ length prefix of the given k-window in the training sequences.

A *2-D frequency profile* for a test sequence can be constructed as follows. First, all k-windows from the test sequence are extracted and the associated tuples (f_k, f_{k-1}) are computed from the training sequences. The f_k frequencies are binned into p bins in the same manner as the 1-D frequency profiles. Similarly, the f_{k-1} frequencies are binned into p bins. Thus, every tuple (f_k, f_{k-1}) is assigned to a "cell" (or grid) on a $p \times p$ grid. The values in each cell are normalized to lie between 0 and 1 by dividing them by the total number of windows in the given sequence. Thus each test sequence can be mapped into a $\Re^{p\times p}$ space.

Note that the column aggregation of the *2-D frequency profile* for a test sequence will give the *1-D frequency profile* for the given test sequence.

Characterizing a Sequence Data Set Using Average 2-D Frequency Profiles

To characterize a given sequence data set, using the 2-D frequency profiles, we follow the same procedure as for 1-D frequency profiles. The frequency profiles for normal and anomalous sequences are aggregated separately to obtain an average normal 2-D frequency profile and an average anomalous 2-D frequency profile, respectively.

A test sequence data set can be characterized with respect to a normal data set by taking the difference between the average 2-D frequency profiles for normal and anomalous test sequences. We will describe how this characteristic can be used to explain the behavior of Markovian techniques in Section 6.2.

6.1.3 Average Sequence Similarity

The third transformation of sequences is motivated from the kernel based techniques which utilize the similarity between a test sequence and the normal sequences to assign an anomaly score to the test sequence.

Let K denote the kernel matrix for a test data set $\mathbf{S}$ computed using (5.3) (See Section 5.1). Let $\tilde{K}$ correspond to row sorted version of K, such that i^{th} row of $\tilde{K}$ consist of the similarity between $S_i \in \mathbf{S}$ and training sequences in $\mathbf{T}$ sorted in increasing order. For a given test sequence S_i, the average of the i^{th} row of $\tilde{K}$ is defined as a separability statistic, also referred to as *average sequence similarity.*

Characterizing a Sequence Data Set Using Average of Average Sequence Similarity

A given test sequence data set can be characterized using the average sequence similarity statistic by computing the average of the average sequence similarity for the normal test sequences and anomalous test sequences. We use the difference between these two quantities as another characteristic for the test sequence data set and show how the performance of kernel based techniques can be explained using it in Section 6.3.

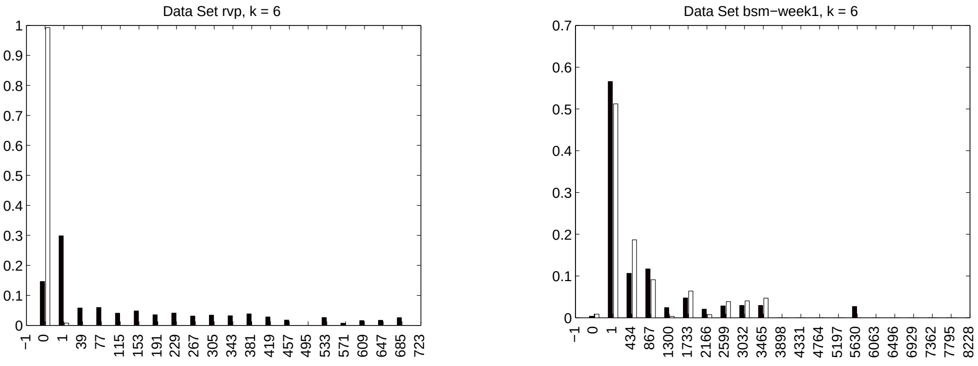
6.2 Relationship Between Performance of Techniques and Frequency Profiles

In this section we relate the performance of the window based (*tSTIDE*) and Markovian techniques (*FSA*, *FSAz*, *PST*, and *RIPPER*) to the 1-D and 2-D frequency profiles defined in Section 6.1.1.

6.2.1 *tSTIDE*

The performance of *tSTIDE* can be explained using the 1-D frequency profiles described in the previous section. The anomaly score assigned by *tSTIDE* is inversely proportional to the frequency of the k-windows in a given sequence. Hence the difference in the 1-D frequency profiles for normal and anomalous test sequences determines the relative performance of *tSTIDE* on a given test data set.

For example, the average 1-D frequency profiles for rvp data set in Figure 6.1(a) are significantly different, and hence the performance of *tSTIDE* is 90% (See Table 5.3). For bsm-week1 data set in Figure 6.1(b), the difference is not significant, and hence the performance of *tSTIDE* is relatively poor (=20%).



(a) rvp

(b) bsm-week1

Figure 6.1: Average 1-D frequency profiles for 6-windows.

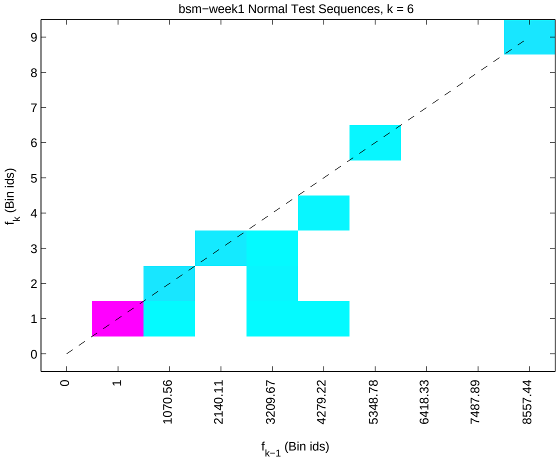
6.2.2 FSA

The $tSTIDE$ technique distinguishes between normal and anomalous test sequences in terms of the frequency of the k-windows, f_k. Often, f_k alone is not distinguishing enough (see Figure 6.1(b)). The FSA technique addresses this issue by considering the frequency of a k-window as well as the frequency of the $k-1$ length suffix of the k-window.

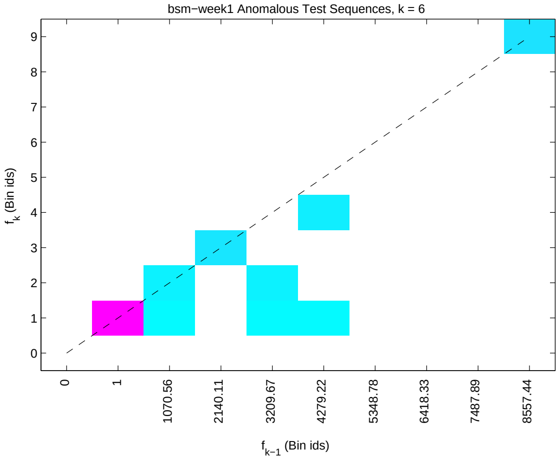
The performance of FSA can be explained using the 2-D frequency profiles described in previous section. FSA assigns anomaly score to a sequence using the values f_k and f_{k-1} for every k window. Hence the difference in the average 2-D frequency profiles for normal and anomalous sequences determines it relative performance on the given data set.

For example, the average 2D frequency profiles for the bsm-week1 data set are shown in Figures 6.2(a) and 6.2(b) for normal and anomalous sequences, respectively. The color of each cell represents the magnitude of the relative proportion of k-windows falling in that cell. We compare the two profiles with the 1D frequency profiles shown in Figure 6.1(b). The absolute difference between normal and anomalous frequency profiles is shown in Figure 6.2(c) with marker "+" indicating that normal test sequences had higher value for that cell than the anomalous test sequences, and marker "△" indicating that normal test sequences had lower value for that cell than the anomalous test sequences. Figure 6.3 shows the plots (differences only) for other public data sets. Note that if the 2D profiles are collapsed onto the y-axis, we will get the corresponding 1D profiles. We note that even though the normal and anomalous sequences are not differentiable when only f_k is considered, the difference is significant when both f_k and f_{k-1} are considered. This is the reason why FSA performs better than $tSTIDE$ on the bsm-week1 data set.

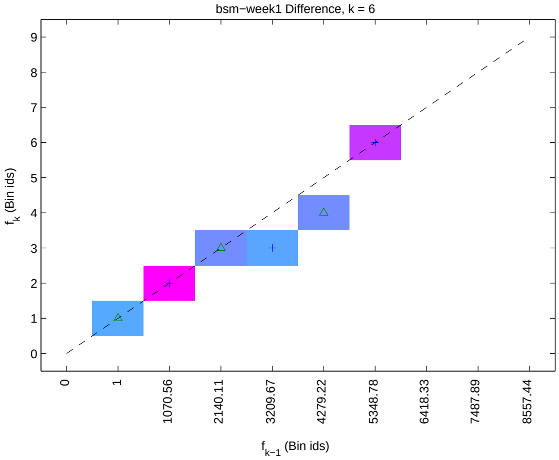
Comparing $tSTIDE$ and FSA The key distinction between $tSTIDE$ and FSA is that the former technique makes use of the frequencies of k-windows while the latter makes use of the frequencies of k-windows and the frequencies of their $k-$ length suffixes. This distinction is illustrated in Figure 6.4 which shows the scores assigned by $tSTIDE$ and FSA to windows, $w(f_k, f_{k-1})$. These scores are also referred to as likelihood scores and are the inverse of the anomaly score of the windows. Since $f_k \leq f_{k-1}$, the entries above the lower diagonal are ignored.



(a) Normal Test Sequences.

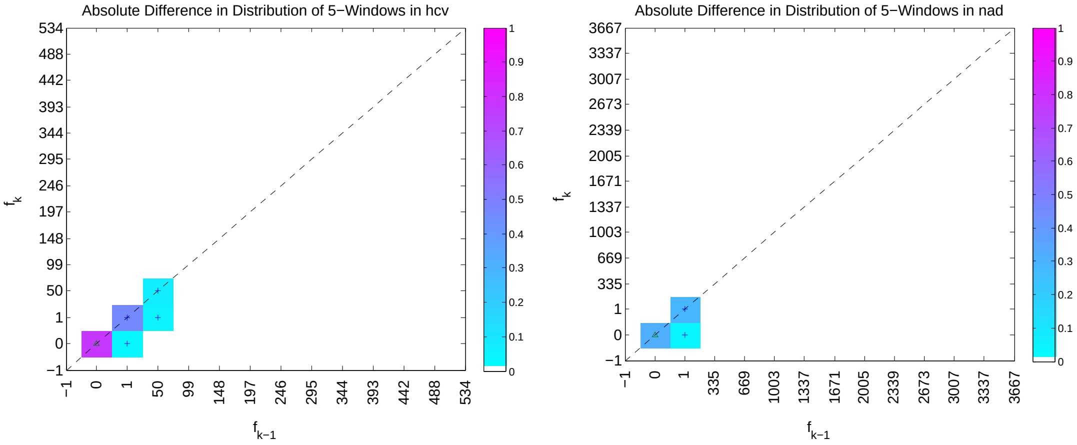


(b) Anomalous Test Sequences.



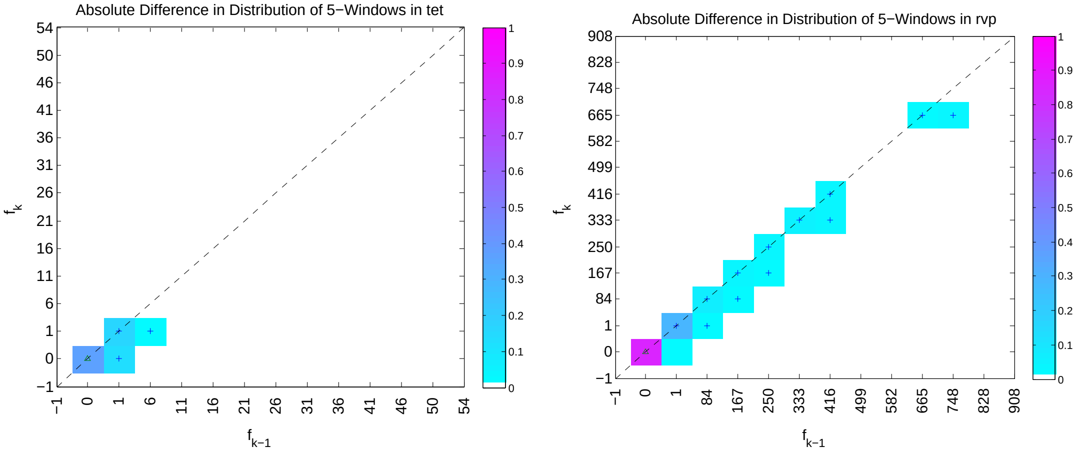
(c) Difference.

Figure 6.2: 2D Average frequency profiles for bsm-week1 data set ($k = 6$).



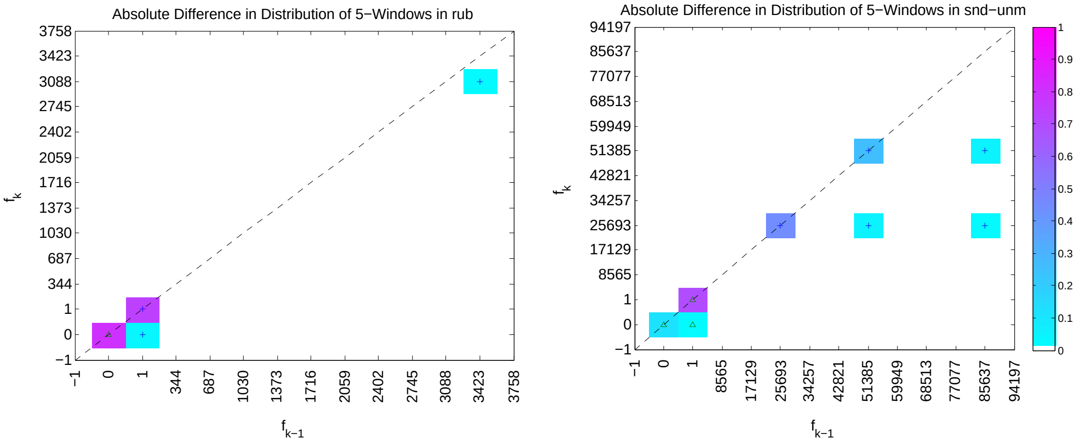
(a) hcv ($\Delta = 1.56$)

(b) nad ($\Delta = 0.64$)



(c) tet ($\Delta = 0.73$)

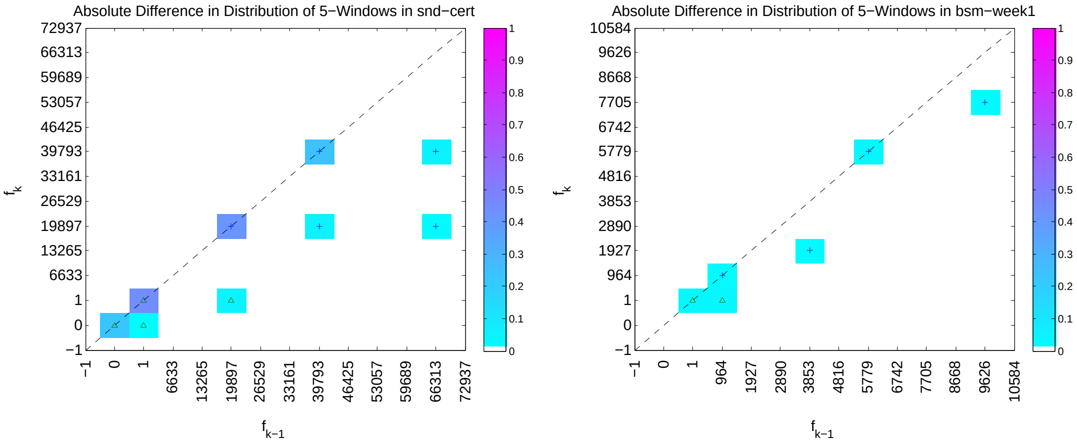
(d) rvp ($\Delta = 1.71$)



(e) rub ($\Delta = 1.62$)

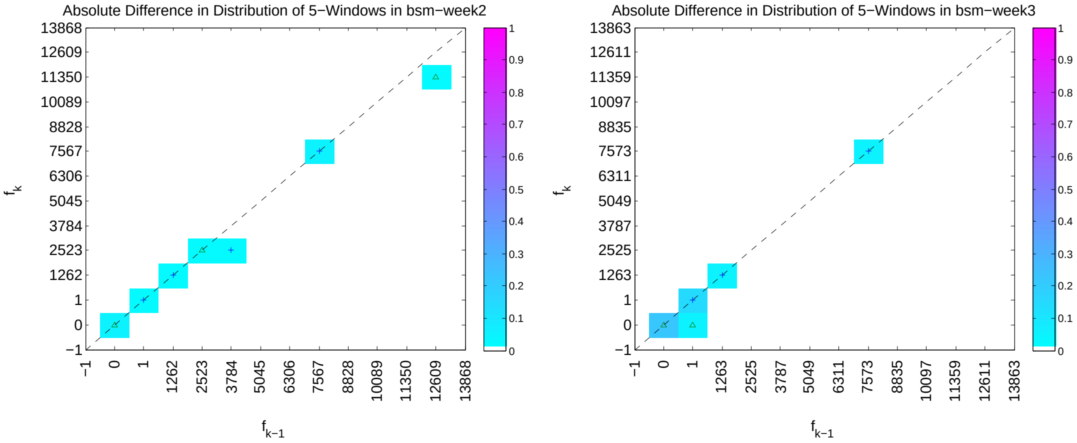
(f) snd-unm ($\Delta = 1.68$)

Figure 6.3: Absolute difference in 2D frequency profiles for public data sets ($k = 6$).



(a) snd-cert (Δ = 1.54)

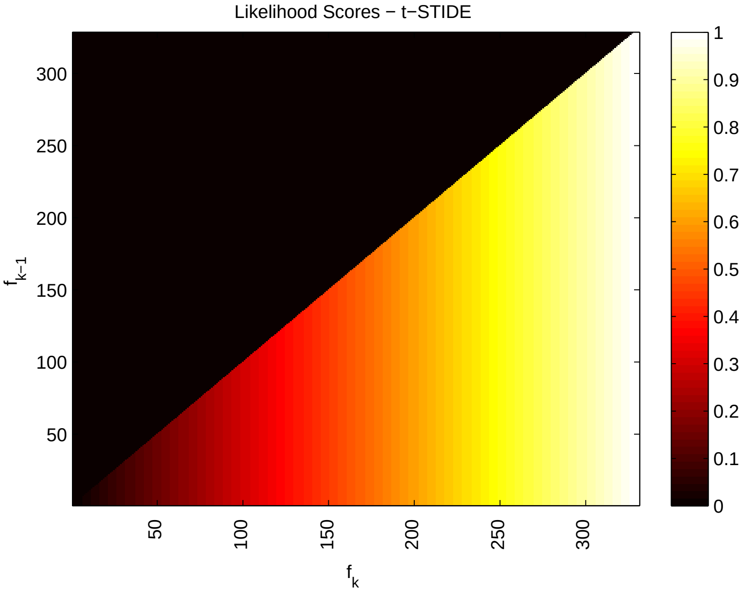
(b) bsm-week1 (Δ = 0.23)



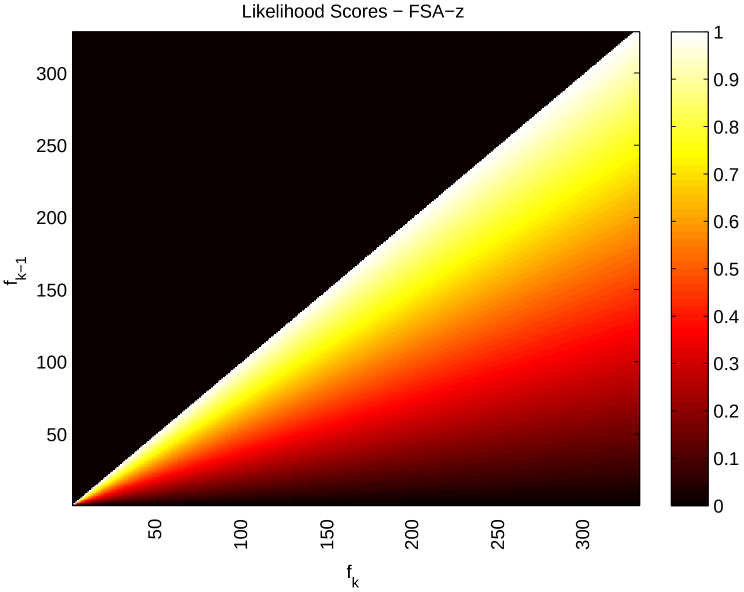
(c) bsm-week2 (Δ = 0.27)

(d) bsm-week3 (Δ = 0.61)

Figure 6.3: Absolute difference in 2D frequency profiles for public data sets ($k = 6$). *Contd ...*



(a) $tSTIDE$.



(b) $FSAz$

Figure 6.4: Likelihood scores $L(f_k, f_{k-1})$, assigned by different techniques.

FSA ignores the k-windows for which $f_{k-1} = 0$, i.e., the bottom left corner of Figure 6.4(b). It is clear that the scores assigned by $tSTIDE$ are independent of f_{k-1} and are linearly proportional to f_k. Thus only high frequency windows will be assigned a high likelihood score by $tSTIDE$. But for FSA, k-windows with low f_k can still be assigned a high score, if the corresponding value of f_{k-1} is also low. This key difference accounts for a key strength and weakness of $tSTIDE$ and FSA.

Consider a scenario in which the training data set is not pure but contains one anomalous sequence, such that most of the k-windows (for a given value of k) do not occur in any other training sequence. Let there be a truly anomalous sequence in the test data set which is similar to the one anomalous training sequence. Most of the k-windows extracted from this test sequence will have $f_k = 1$. $tSTIDE$ will assign a high anomaly score to this test sequence. Though the value of f_{k-1} cannot be guaranteed, it is likely that $f_{k-1} \approx 1$. Thus FSA will assign a high likelihood score to the k-windows of the anomalous test sequence, and hence assign it a low anomaly score. Thus $tSTIDE$ is a better technique in this scenario.

Now consider a different scenario, in which the training data set contains a sequence that consists of k-windows that do not occur in any other training sequence, but are normal. Let the test data contain one truly normal sequence similar to this training sequence. $tSTIDE$ will assign a high anomaly score to this test sequence because the windows extracted from this sequence will have $f_k = 1$. But, similar to the argument for the previous scenario, FSA will assign a low anomaly score. Thus FSA is a better technique in this scenario.

To summarize, $tSTIDE$ is more robust when the training data might not be pure, i.e., it might contain anomalous sequences. $FSAz$ is a better choice when the training data has rare but normal patterns (windows) that have to be learnt.

6.2.3 $FSAz$

One issue with FSA is that it ignores the k-windows for which $f_k = f_{k-1} = 0$. But often, such windows can differentiate between normal and anomalous sequences. The plots of differences between the 2D average frequency profiles for normal and anomalous sequences, shown in Figure 6.3, show that for several data sets, anomalous test sequences have a higher proportion of such windows than the normal data sets. Our proposed technique, $FSAz$, utilizes this information by assigning a likelihood score of 0 to such sequences, instead of ignoring them. This makes $FSAz$ perform better than FSA for most data sets.

6.2.4 PST

An issue with FSA (and $FSAz$), as noted earlier, is that they estimate the conditional probability of a symbol, based on its fixed length history, even if the history occurs once in the training sequences. Thus such estimates can be unreliable, and hence make the techniques highly susceptible to presence of anomalies in the training set. PST addresses this issue by conditioning the probability of a symbol on its k length history, only if the history occurs a significant number of times in the training sequences. If the frequency of the history is low, i.e., the conditional probability estimate is unreliable, it uses the longest suffix of the history which satisfies the reliability threshold.

The score assigned by PST to a k-window is lower bounded by the score assigned by $FSAz$. The actual score assigned by PST not only depends on f_k, f_{k-1}, but also on δ, which is a threshold on f_{k-1}. The value of δ is determined using user-defined parameters and the training sequences (See Section 5.1.3). For a given k-window, if $f_{k-1} \geq \delta$ the score assigned by PST is same as $FSAz$. Otherwise, PST chooses the longest suffix of the k window of length j $(2 \leq j \leq k)$, such that $f_{j-1} \geq \delta$. If $f_1 < \delta$, PST assigns the score equal to the probability of observing the last (k^{th}) symbol of the given window.

For example, Figures 6.5(a)–6.5(e) show the difference in the frequency profiles for normal and anomalous test sequences for the rub data set for different values of k. To assign a score to a k-window for $k = 6$, the PST technique will first consider the frequency profile for $k = 6$. Let us assume that the k-window to be scored has $f_{k-1} < \lambda$. In this case PST will substitute the score with the score of a window of length $k - 1$ in the frequency profile for $k - 1$ length windows. If for the $k - 1$ window, $f_{k-2} \geq \lambda$, the corresponding score will be used, otherwise the frequency profile for $k - 3$ is considered, and so on.

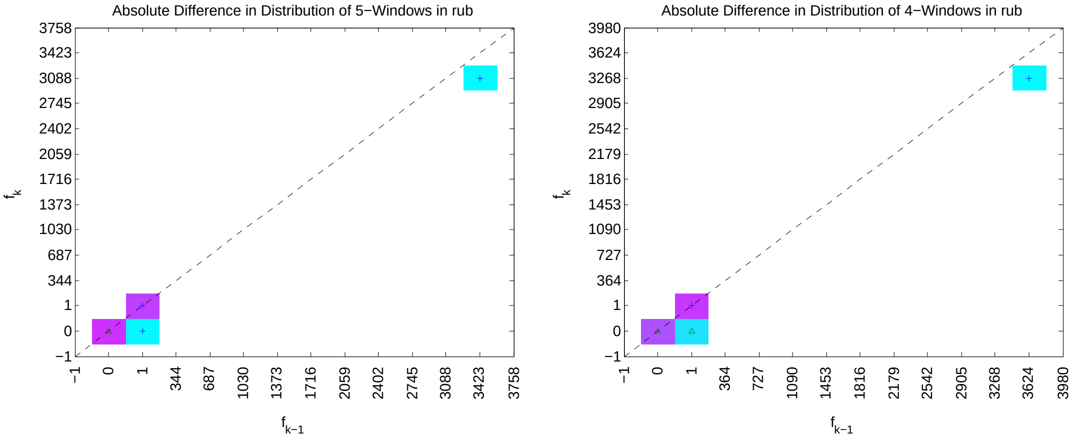
Using this understanding of PST, we can explain why PST performs significantly poorly than $FSAz$ for most of the public data sets. Let us consider the data set rub. Figure 6.5(a) shows difference in the frequency profiles of normal and anomalous test sequences for $k = 6$. The distinguishing cells in the profile are mostly located in the bottom left corner, and there is a single distinguishing cell in the upper right corner. Both PST and $FSAz$ will assign similar scores to the k-windows belonging to the cell in the upper right corner. For the cell in the bottom leftmost corner, $FSAz$ will assign a 0 score, and for other cells $FSAz$ will assign a higher score. Thus $FSAz$ will be able to distinguish between normal and anomalous test sequences, which supports our experimental finding that the performance of $FSAz$ on this data set is (0.88). For PST, all k-windows belonging to the cells in the bottom left corner will have $f_{k-1} < \lambda$, and hence will be substituted with scores for shorter suffix of the k-windows. Thus the scores for windows to the bottom leftmost cell in Figure 6.5(a) will be scored same as the shorter windows (of length $j < k$) in Figures 6.5(b)–6.5(e) for which $f_{j-1} \geq \delta$. But it is evident from the plots that normal and anomalous test sequences are not significantly distinguishable for higher values of f_{j-1}. This is reason why PST performs poorly for this data set (0.28).

While the above mentioned behavior of PST is an obvious disadvantage for most of the data sets, it can also favor PST in certain cases. For example, PST performs well in comparison to $FSAz$ on the artificial data set $d6$. The frequency profiles of normal and anomalous test sequences for $d6$ are shown for different values of k in Figures 6.6(a) – 6.6(e). We observe that the frequency profiles for normal and anomalous test sequences are not distinguishable for $k = 6$ and hence $FSAz$ performs poorly (0.38). But when frequency profiles for lower values of $j \leq 3$ are considered by the PST, the profiles for normal and anomalous sequences are relatively more distinguishable (even for larger values of f_{j-1}) and hence PST performs better (0.68).

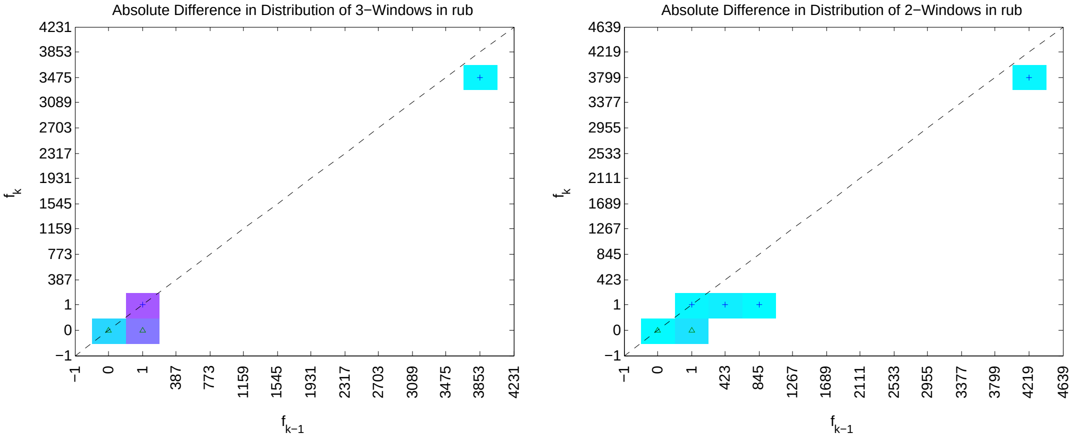
6.2.5 $RIPPER$

The motivation behind $RIPPER$ is same as PST, i.e., if the fixed length history of a symbol in a test sequence does not have a reliable frequency in the training sequences, the symbol is conditioned on a subset of the history. The difference being that the subset is not the suffix of the history (as is the case with PST), but a subsequence of the history.

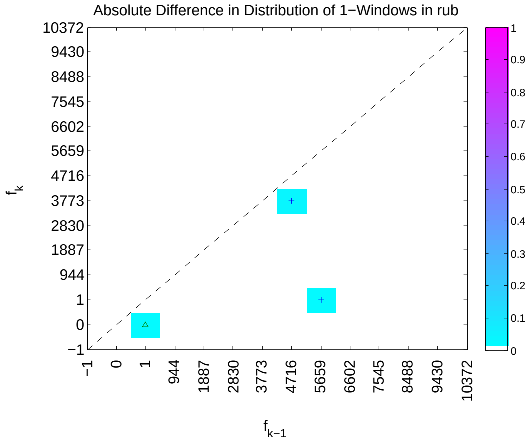
$RIPPER$, like PST, assigns score to a k-window which is lower bounded by the score assigned by $FSAz$. The actual score assigned by PST depends on the $RIPPER$



(a) $k = 6$ ($\Delta = 1.62$) (b) $k = 5$ ($\Delta = 1.62$)

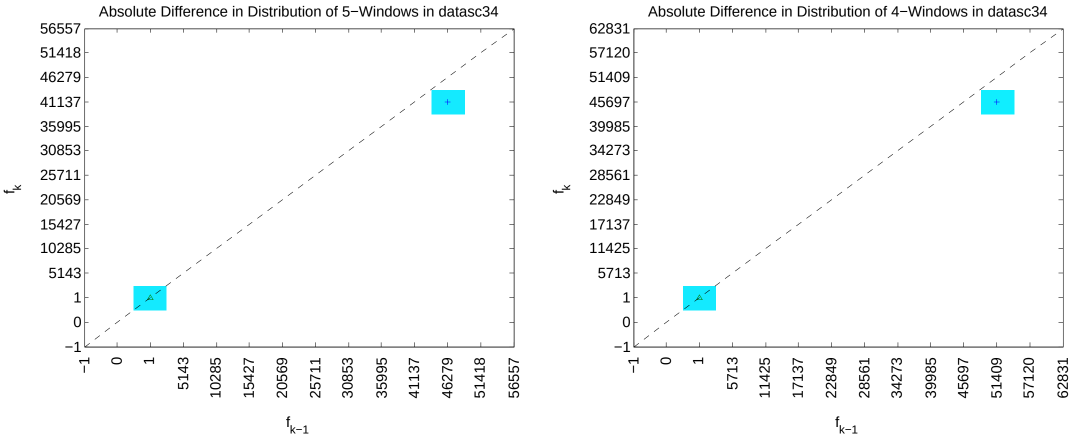


(c) $k = 4$ ($\Delta = 1.37$) (d) $k = 3$ ($\Delta = 0.32$)

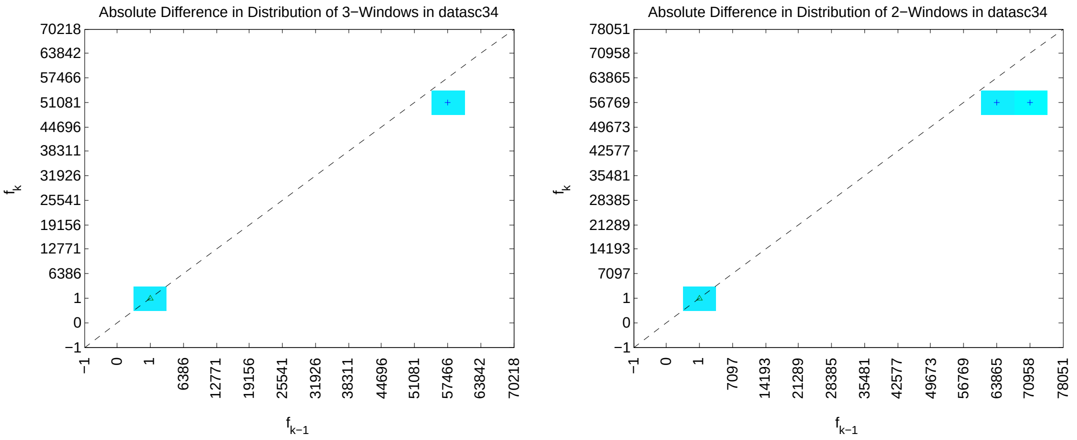


(e) $k = 2$ ($\Delta = 0.13$)

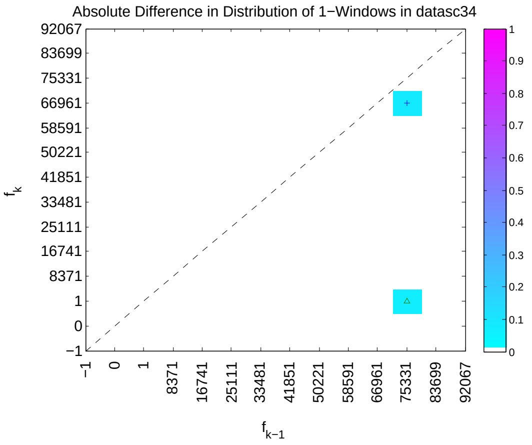
Figure 6.5: Absolute difference in frequency profiles for rub data set.



(a) $k = 6$ ($\Delta = 0.19$) (b) $k = 5$ ($\Delta = 0.19$)



(c) $k = 4$ ($\Delta = 0.20$) (d) $k = 3$ ($\Delta = 0.19$)



(e) $k = 2$ ($\Delta = 0.16$)

Figure 6.6: Absolute difference in frequency profiles for $d6$ data set.

rule that is “fired” for the $k-1$ prefix of the given window. If the target of the fired rule matches the k^{th} symbol of the given window, the likelihood score is 1, else the likelihood score is the inverse of the confidence associated with the rule. It is difficult to analytically estimate the actual scores assigned by $RIPPER$, but generally, the scores assigned by $RIPPER$ are higher than $FSAz$ but lower than PST.

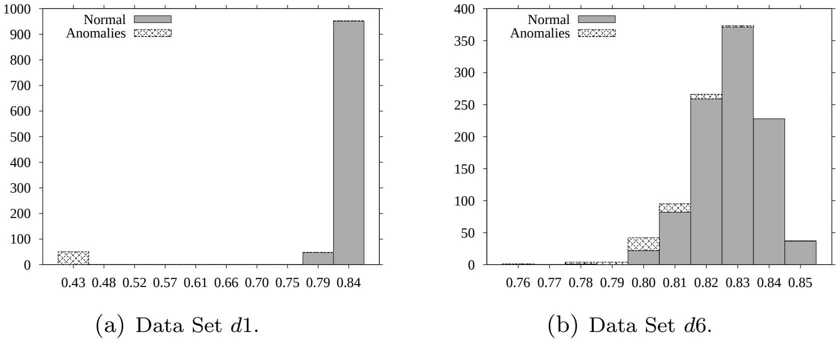
As mentioned earlier, the scores assigned by $RIPPER$ are same as $FSAz$ for higher values of f_k. For lower values of f_k the scores depend on the distribution of k-windows in the training data set as well as how the underlying classifier ($RIPPER$) learns the rules and what is the order in which the rules are applied. Generally speaking, it can be stated that the scores assigned by $RIPPER$ to such windows is greater than 0 but lower than the score assigned by PST to such windows.

The above mentioned behavior of $RIPPER$ results in its poor performance in cases in which the cells with lower values of f_k are distinguishing and the anomalous test sequences have higher proportion of windows in that cell than the normal test sequences. $RIPPER$ assigns a higher overall likelihood score to the anomalous test sequences and hence is not able to distinguish them from normal sequences. For all PFAM data sets the distinguishing cells have lower f_k value, resulting in poor performance of $RIPPER$. For UNM data sets, the distinguishing cells have higher values for f_k and hence the performance of $RIPPER$ is very close to that of $FSAz$.

6.3 Impact of Nature of Similarity Measure on Performance of Anomaly Detection Techniques

Kernel based techniques (kNN and CLUSTER) are distinct from the window based and Markovian techniques because they rely on the similarity between a test sequence and training sequences to assign anomaly score to the test sequence. Thus their performance can be explained using the *average similarity to training sequences characteristic*, as described in Section 6.1.3.

One distinction between normal and anomalous sequences is that normal test sequences are expected to be more similar (using a certain similarity measure) to training sequences, than anomalous test sequences. If the difference in similarity is not large, this characteristic will not be able to accurately distinguish between normal and anomalous sequences. This characteristic is utilized by kernel based techniques (kNN and CLUSTER) to distinguish between normal and anomalous sequences.



(a) Data Set $d1$.

(b) Data Set $d6$.

Figure 6.7: Histogram of Average Similarities of Normal and Anomalous Test Sequences to Training Sequences.

For example, Figure 6.7(a) shows the histogram of the average ($nLCS$) similarities of test sequences in the artificial data set $d1$ to the training sequences. The normal test sequences are more similar to the training sequences, than the anomalous test sequence. This indicates that techniques that use similarity between sequences to distinguish between anomalous and normal sequences will perform well for this data set. From Table 5.5, we can observe that the performance of CLUSTER as well as kNN is 100% on $d1$. A similar histogram for data set $d6$ is shown in Figure 6.7(b), which shows that average similarities of normal test sequences and the average similarities of anomalous test sequences are very close to each other. This confirms the observation in Table 5.5 that CLUSTER and kNN should perform poorly for this data set.

We quantify the above characteristic by computing the average sequence similarity for each test sequence. Let the average of the average similarities for normal test sequences be denoted as s_n, and average of the average similarities for anomalous test sequences be denoted as s_a. If for a given data set, the difference $s_n - s_a$ is large, kNN and CLUSTER are expected to perform well on that data set, and vice-versa.

Tables 6.1 and 6.2 show the values of s_n, s_a, and $s_n - s_a$, for the real and artificial data sets, respectively. The performance of both kNN and CLUSTER is highly correlated with the difference $s_n - s_a$.

	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3
s_n	0.53	0.48	0.67	0.82	0.75	0.99	0.99	0.97	0.98	0.97
s_a	0.38	0.38	0.37	0.36	0.37	0.50	0.38	0.88	0.81	0.73
$s_n - s_a$	0.15	0.10	0.30	0.46	0.38	0.49	0.61	0.09	0.17	0.24

Table 6.1: Values of s_n, s_a for the public data sets.

	d1	d2	d3	d4	d5	d6
s_n	0.87	0.87	0.86	0.86	0.86	0.86
s_a	0.45	0.63	0.63	0.73	0.76	0.78
$s_n - s_a$	0.42	0.24	0.23	0.13	0.10	0.08

Table 6.2: Values of s_n, s_a for the artificial data sets.

6.4 Using RBA Features for Anomaly Detection

A key aspect of the RBA framework is that it maps data instances into a multivariate continuous space, where normal and anomalous instances can be distinguished from each other. Thus, applying the RBA framework is equivalent to extracting features from the sequence data set. In this section, we propose two novel techniques based on these features to detect anomalies in a given test data set. We denote the novel techniques as WIN_{1D} and WIN_{2D}, since they utilize the 1-D and 2-D frequency profiles discussed in Sections 6.1.1 and 6.1.2, respectively.

The motivation behind these two techniques is the fact that several existing techniques implicitly utilize the difference between the relative frequencies of the k-windows to distinguish between normal and anomalous sequences (See Section 6.2).

The algorithm for the first technique, WIN_{1D}, is as follows:

$WIN_{1D}(k, p, nn, \mathbf{S}, \mathbf{T})$

1. For each training sequence $T_j \in \mathbf{T}$, calculate its 1-D frequency profile with respect to $\mathbf{T}$ (denoted as $\tilde{T_j}$) with window size k and number of bins as p.
2. For each test sequence $S_i \in \mathbf{S}$, calculate its 1-D frequency profile with respect to $\mathbf{T}$ (denoted as $\tilde{S_i}$) with window size k and number of bins as p.
3. For each "mapped" test sequence, $\tilde{S_i}$, calculate its anomaly score as equal to the distance to its nn^{th} nearest neighbor in $\tilde{\mathbf{T}}$[1] using *Euclidean* distance

---

[1] $\tilde{\mathbf{T}}$ is the set of "mapped" training sequences using the 1-D frequency profiles.

metric.

The algorithm for the second technique, WIN_{2D}, is as follows:

$WIN_{2D}(k, p, nn, \mathbf{S}, \mathbf{T})$

1. For each training sequence $T_j \in \mathbf{T}$, calculate its 2-D frequency profile with respect to $\mathbf{T}$ (denoted as $\tilde{\tilde{\mathrm{T}}}_j$) with window size k and number of bins as p.
2. For each test sequence $S_i \in \mathbf{S}$, calculate its 2-D frequency profile with respect to $\mathbf{T}$ (denoted as $\tilde{\tilde{\mathrm{S}}}_i$) with window size k and number of bins as p.
3. For each "mapped" test sequence, $\tilde{\tilde{\mathrm{S}}}_i$, calculate its anomaly score as equal to the distance to its nn^{th} nearest neighbor in $\tilde{\tilde{\mathbf{T}}}$[2]using *Euclidean* distance metric.

The inputs to both techniques are the training data set, $\mathbf{T}$, test data set, $\mathbf{S}$, window size, $k(\geq 2)$, number of bins, $p(\geq 2)$, and number of nearest neighbors to analyze, nn.

6.4.1 Results on Public and Artificial Data Sets

We evaluate the performance of the proposed techniques on the public and artificial data sets described in Chapter 5.

Sensitivity to Parameters

We first investigate the sensitivity of the different parameters on the performance of WIN_{1D} and WIN_{2D}. The techniques gave best overall performance for window size $k = 6$, which was also the best performing window size for the existing window based and Markovian techniques. The performance of both techniques was not sensitive to the number of nearest neighbors. The techniques gave best performance when the number of bins used to construct the profile, p, was low (≈ 3). For larger values of p, the dimensionality of the mapped data increased, and hence the performance of the distance based anomaly detection technique deteriorated.

For the results provided in subsequent section, the optimal parameter settings were found by testing on a validation set for different combinations of the parameters (p, k, nn)

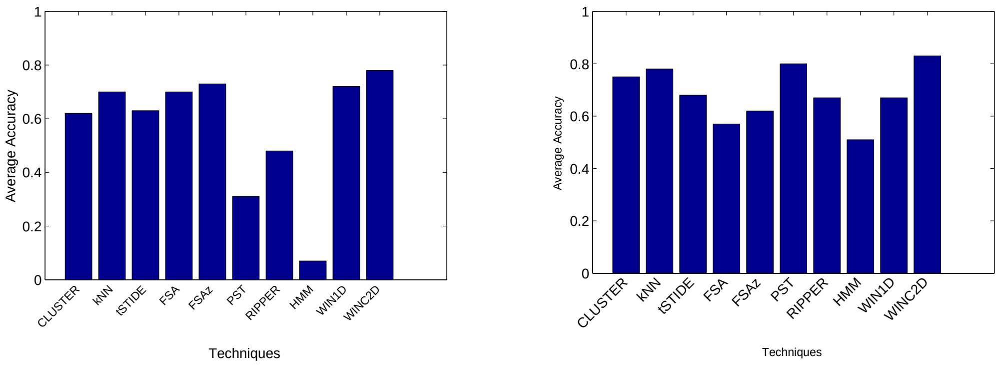
---

[2] $\tilde{\tilde{\mathbf{T}}}$ is the set of "mapped" training sequences using the 2-D frequency profiles.

and using the combination the provides best average results across all data sets. The results are shown for $p = 5$, $k = 6$, and $nn = 5$.

Comparison with All Existing Techniques

The first set of results show how WIN_{1D} and WIN_{2D} compare against the state of art techniques, discussed in Chapter 5, on the different public and artificial data sets. The comparison of average performance of the proposed techniques with the existing techniques is shown in Figure 6.8. Notably, WIN_{1D}, which is based on $tSTIDE$, shows better performance than $tSTIDE$, and WIN_{2D}, which is based on $FSAz$, shows better performance than $FSAz$ on both public and artificial data set. Overall, WIN_{2D} performs significantly better than all existing techniques on average across all public and artificial data sets.



(a) Public Data Sets

(b) Artificial Data Sets

Figure 6.8: Comparison of average accuracies for WIN_{1D} and WIN_{2D}, and existing anomaly detection techniques.

The reason the proposed techniques perform better than the existing techniques is because of the way the windows are utilized by the proposed techniques. For example, let us consider $tSTIDE$ and WIN_{1D}. Both of these techniques use the frequency of k-windows to distinguish between the normal and anomalous test sequences. But $tSTIDE$ weights the windows in a test sequence by their frequencies and the sums the total weights to get an inverse of the anomaly score. On the other hand, WIN_{1D} bins the windows based on their frequency, and then uses the normalized bin counts as features. By using a nearest neighbor approach, WIN_{1D} "learns" weights on different windows to achieve best separability between the normal and anomalous test sequence. Same holds true for $FSAz$ and WIN_{2D}.

Comparison with Best Existing Technique

The strength of the RBA based techniques, WIN_{1D} and WIN_{2D}, is that they distinguish between normal and anomalous test sequences in a multi-dimensional space, while most of the existing techniques operate along one or a limited subset of the dimensions. In the second set of results we assess if this strength allows the RBA based techniques to outperform the existing techniques.

In Table 6.3, we compare the accuracy results for WIN_{1D} and WIN_{2D} against the best existing technique for each public data set. The results show that both WIN_{1D} and WIN_{2D} are strictly better or comparable with the best existing technique for almost all of the public data sets. The same inference can be drawn from the AUC results for the public data sets in Table 6.4.

For artificial data sets, the performance of WIN_{2D} is still significantly better than the best existing technique for each data set as shown in Tables .6.5 and 6.6. Notably, for the artificial data sets, the performance of WIN_{1D} is relatively worse than the best existing technique.

For artificial data sets, PST was found to be the best technique while both FSA and $FSAz$ were found to perform poorly for many artificial data sets (See Chapter 5, Section 5.3). The reason was that the artificial data sets were designed to break FSA and $FSAz$, while PST, which utilizes the frequencies of varying length suffixes of the k length windows, was able to distinguish between the normal and anomalous test sequences. By using WIN_{2D}, the behavior of PST is captured and improved, and hence WIN_{2D} outperforms PST on the artificial data sets.

6.5 Conclusions and Future Work

In this chapter we showed how the RBA framework can be used in the context of anomaly detection for symbolic sequences. Visualizing symbolic sequences is challenging, especially when the sequences are of varying length. Using the RBA framework we

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
WIN_{1D}	0.92	0.74	0.52	0.90	0.88	0.82	0.88	0.30	0.60	0.66	0.72
WIN_{2D}	0.92	0.76	0.82	0.92	0.92	0.84	0.88	0.50	0.60	0.66	0.78
Existing Best	0.92	0.74	0.50	0.90	0.88	0.82	0.88	0.50	0.56	0.66	0.74

Table 6.3: Comparing accuracy of WIN_{1D} and WIN_{2D} against best existing technique for public data sets.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
WIN_{1D}	1.00	0.98	0.98	1.00	1.00	0.99	0.98	0.75	0.92	0.92	0.95
WIN_{2D}	1.00	0.99	0.99	1.00	1.00	0.99	0.98	0.91	0.93	0.92	0.97
Existing Best	1.00	0.98	0.98	1.00	1.00	0.99	0.96	0.88	0.91	0.97	0.97

Table 6.4: Comparing AUC of WIN_{1D} and WIN_{2D} against best existing technique for public data sets.

	d1	d2	d3	d4	d5	d6	**Avg**
WIN_{1D}	1.00	0.92	0.58	0.52	0.34	0.64	0.67
WIN_{2D}	1.00	0.96	0.81	0.76	0.71	0.74	0.83
Existing Best	1.00	0.84	0.82	0.76	0.68	0.68	0.80

Table 6.5: Comparing accuracy of WIN_{1D} and WIN_{2D} against best existing technique for artificial data sets.

	d1	d2	d3	d4	d5	d6	**Avg**
WIN_{1D}	1.00	1.00	0.91	0.89	0.76	0.87	0.90
WIN_{2D}	1.00	1.00	0.98	0.98	0.96	0.98	0.98
Existing Best	1.00	0.96	0.98	0.98	0.96	0.95	0.97

Table 6.6: Comparing AUC of WIN_{1D} and WIN_{2D} against best existing technique for artificial data sets.

provide a visualization scheme for symbolic sequences.

The RBA based mapping for the symbolic sequences is motivated from the existing techniques that use fixed length windows as a unit of analysis. In this chapter we have shown how, using the RBA based features, one can understand the performance of the different existing techniques. Moreover, the framework can also be used to identify the fundamental differences between techniques. For example, $tSTIDE$ and FSA are shown to be highly different from each other since they handle k-windows in distinct manner. The framework also allows to identify the weaknesses of each technique. For example, the poor performance of PST on most of the real data sets could be explained using the framework. The same framework can also be used to construct scenarios in which a given technique would perform well or poorly.

The analysis of the various distinguishing characteristics can also aid in choosing optimal values of parameters for different techniques. For example, Figure 6.5 shows the magnitude of difference between normal and anomalous sequences in rub data set for different values of window size k. The maximum difference occurs when $k = 5$ or 6. Our results indicate that all techniques that depend on window size as a parameter give optimal performance for these values of k. Similarly, for kNN and $CLUSTER$, the difference in the corresponding characteristic for normal and anomalous test sequence, can be calculated for different values of the parameter k. The value of k that results in maximum difference in terms of the characteristic, is likely to give best performance on that data set. One could argue that given a labeled validation data set, a technique can be evaluated for different parameter values to obtain the optimal value. But using the proposed framework, the analysis needs to be done only for a characteristic, without having to test every technique that depends on that characteristic.

The most significant outcome of applying the RBA framework to symbolic sequences is that the features obtained from the mapping can be used to develop powerful anomaly detection techniques, which outperform the existing techniques. Moreover, the RBA based techniques are shown to better than the best existing technique for most of the data sets. Thus instead of using different existing techniques which are optimal for different data sets, RBA provides one best technique across a variety of data sets. This is a significant step towards the ultimate goal for the anomaly detection research, which is to find a technique that can perform well across all application domains.

Part III

Detecting Anomalies in Time Series Data

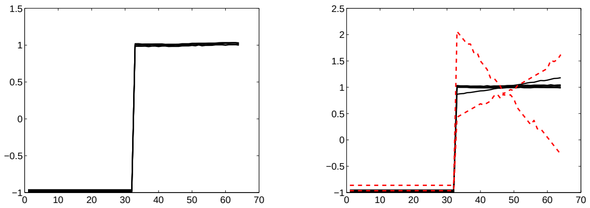
Chapter 7

Detecting Anomalies in a Time Series Database

In this chapter, we investigate the problem of detecting anomalies in a given time series, with respect to a reference set consisting of normal time series. This problem formulation is highly applicable in domains of aircraft health management [165], credit card fraud detection [55], detecting abnormal conditions in ECG data [119], detecting shape anomalies [183], detecting outlier light curves in astronomical data [146, 191], etc.

For example, in aircraft health management, during an aircraft's flight, multiple sensors measure flight parameters that indicate system health. This data is collected as time series. A fault in the aircraft's flight, such as failure of a component, is manifested as anomalies in one or more of the sensor readings generated by the aircraft. Figure 7.1(a) shows a set of reference time series corresponding to measurements from a healthy rotary engine disk of an aircraft, and Figure 7.1(b) shows a test set of time series corresponding to measurements from healthy (solid) and cracked (dashed) disks. Detecting when an engine disk develops cracks is crucial. This task requires finding anomalies in the test time series. Online detection of these anomalies allows preventive measures that can save lives. Off-line detection of the anomalies is critical for fault diagnosis.

Many anomaly detection techniques that solve the above mentioned problem for time series data have been proposed, such as kernel based techniques [146, 191, 183]



(a) Reference (b) Test

Figure 7.1: Reference and test time series for the NASA disk defect data [164].

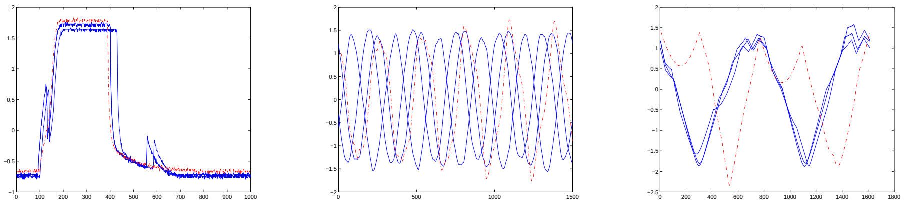
and segmentation techniques [156, 128, 32]. In addition, a number of techniques have been proposed for related problems. For example, Keogh et al [101] have developed a number of techniques for discord detection, where discords are defined as unusual subsequences in a long time series [100, 103, 62, 27, 191]. Ma and Perkins [124] have proposed a technique to detect anomalous observations in a long time series.

However, most of these techniques have been studied in the context of specific domains, such as detecting faults in operational data [156, 128, 32], detecting outlier light curves in astronomical data [146, 191], and detecting shape anomalies[1] [183]. While each published study shows the effectiveness of the particular technique in the target domain, there has not been any attempt to analyze the problem in its entirety. The reason such analysis is essential is that the nature of the time series and the nature of anomalies in different domains differs fundamentally. While a technique is shown to be effective for a particular domain, the same technique is not guaranteed to perform well in a different domain where a different type of time series data is encountered.

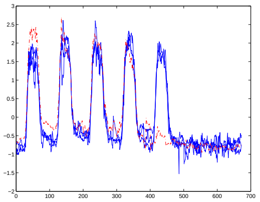
For example, Figure 7.2 shows the normal and anomalous time series plots for four different publicly available data sets taken from varied domains. The *valve* data set corresponds to current measurements recorded on a valve on a space shuttle. The *motor* data set corresponds to functioning of an induction motor. The *shapes* data set corresponds to time series obtained from different physical shapes. The *power* data set corresponds to the weekly power usage by a research plant. We observe that the nature of normal time series as well as anomalous time series is different for the four data sets. For the *valve* data set in Figure 7.2(a), the anomalous time series are mostly

---

[1] The shapes are converted into time series using techniques such as *distance from centroid* [197].



(a) Valve (b) Motor (c) Shapes



(d) Power

Figure 7.2: Normal (blue) and anomalous (red dotted) time series for different data sets [99]. (*Best viewed in color*)

similar to the normal time series except for two short regions. For the *shapes* data set in Figure 7.2(c), the anomalous time series is completely different from the normal time series. The normal time series for the *motor* and *power* data sets in Figure 7.2(b) and 7.2(d), respectively, are periodic. The anomalous time series for the *motor* data set is also periodic but is noisier than the normal time series. The anomalous time series for the *power* data set contains one different cycle. While the normal time series for *valve*, *shapes*, and *power* data sets are similar to each other, the normal time series for the *motor* data set are *out of sync* with each other. Even for this limited sample of data sets, we observe remarkable difference in the nature of normal and anomalous time series. This underlying difference in the data makes the problem of anomaly detection different for each domain. Thus it is highly important to understand how the performance of a technique is related to the nature of the underlying data.

Our objective in this chapter is to get deeper insights into the behavior of different techniques to help us answer the following question – *Which anomaly detection technique is optimal for a given time series data set?* We investigate different ways of solving this problem. First way is to use distance or similarity kernels for time series data. Second is to extract fixed length windows from a test time series and assign anomaly scores to the windows. Third is to learn a predictive or forecasting model from the training data and use the model to detect anomalies in a test time series. Fourth is to learn a state space model for the normal time series and detect anomalies by passing a test time series through the state space model.

We adapt several machine learning techniques (such as one class support vector machines, nearest neighbor density estimation, support vector regression) to detect anomalies in time series data. Our work is novel in the sense that these adaptations have not been tried before for the problem of detecting anomalous time series in a given database of time series data. We evaluate these novel adaptations along with existing state of the art anomaly detection techniques for time series data [191, 32]. One of our novel adaptations, that uses one class SVMs, is shown to perform better than the existing anomaly detection techniques. To understand the performance of existing anomaly detection techniques for symbolic sequences discussed in Chapter 5, we discretized the continuous time series data and applied the symbolic techniques on the discretized data. We evaluate the different techniques on a large variety of time series data sets obtained from a broad spectrum of application domains. The data sets have different characteristics in terms of the nature of normal and anomalous time series. We evaluate the techniques using varied metrics, such as accuracy in detecting the anomalous time series, sensitivity to parameters, and computational complexity. We provide useful insights regarding the relative performance of different techniques based on the experimental evaluation and relate the performance of different techniques to the nature of the underlying time series data.

In Chapter 3 we discussed the related research done in the area of anomaly detection for time series data (See Section 3.5). Only a limited set of techniques have been proposed to solve the semi-supervised anomaly detection techniques, though related techniques have been proposed to detect anomalous observations in a time series [60, 2, 154, 63, 188, 144, 195, 124, 105] or to detect anomalous subsequences in a time series [101, 100, 103, 119, 62, 27, 191]. We have adapted some of these techniques to handle the semi-supervised anomaly detection problem.

The rest of this chapter is organized as follows. Section 7.1 describes various characteristics of time series data which can affect the performance of different anomaly detection techniques. Section 7.2 describes the different anomaly detection techniques for time series data. Section 7.3 describes the different data sets used for evaluating the different techniques. Section 7.4 summarizes our experimental results on the different data sets. We provide conclusions based on the experimental results and the future directions of research in Section 7.5.

7.1 Nature of Input Data

We use the training data, $\mathbf{X}$, to "learn" a model for normal behavior and assign an anomaly score to each test time series, y_j, based on the model. The performance of any technique depends on the nature of normal as well as anomalous time series. The normal time series in the training and test database can vary among themselves due to one or more of following factors:

- The normal time series might contain noise. The noise might be generated as random noise, or a different process.
- The normal time series might not be synchronized, i.e., the data collected for different time series might have different start times. The normal time series in the *motor* data set (Figure 7.2(b)) are examples of such time series.

A key characteristic of the anomaly detection problem is the cause of anomaly within an anomalous time series. Broadly, two types of anomalies can be defined:

Process Anomalies: The generative process for the normal and anomalous time series are completely different, e.g., the anomalous time series in the *shapes* data set (See Figure 7.2(c)) is an example of process anomaly.

Subsequence Anomalies: The normal time series are generated from a single generative process. In the anomalous time series, majority of the observations are generated by the same process, but a few observations are generated by a different process (See Figures 7.2(d) and 7.2(a)). Such anomalous observations are also referred to as *discordant observations* [2] when they occur individually, or as *discords* [101] when they occur as a subsequence.

7.2 Anomaly Detection Techniques for Time Series Data

We investigate following different ways of solving the problem of detecting anomalies in time series data:

1. Kernel Based Techniques These techniques make use of a distance kernel defined for every pair of time series. We evaluate a nearest neighbor based technique, called $KNNC$ (k-nearest neighbor for continuous time series) [191], that utilizes this distance kernel. $KNNC$ assigns an anomaly score to a test time series as equal to its distance to its k^{th} nearest neighbor in the training database, $\mathbf{X}$. We also evaluated a discrete version of $KNNC$, called $KNND$ (where 'D' stands for discrete) [38], where the continuous time series were first discretized into a sequence of symbols, using the Symbolic ApproXimation (SAX) technique [120].

For $KNNC$, we evaluate three distance measures, viz., *Euclidean Distance*, *Dynamic Time Warp* (DTW) [19], and *Cross Correlation* [146].

For $KNND$, we evaluate four similarity measures: *Simple Matching Coefficient* (SMC), *weighted Simple Matching Coefficient* ($wSMC$) using the weight for each pair of symbols [120], *normalized longest common subsequence length* ($nLCS$) [28], and a similarity measure using *time series bitmaps* (BITMAP) [109].

2. Window Based Techniques These techniques extract fixed length (w) windows from a test time series, and assign an anomaly score to each window. The per-window scores are, then, aggregated to obtain the anomaly score for the test time series. The score assignment to a window and the score aggregation can be done in different ways.

We denote the window based techniques for continuous time series as $WINC$, and for discretized sequences as $WIND$. We evaluated two variants of $WINC$: $WINC_{SVM}$ and $WINC_{kNN}$. $WINC_{SVM}$ learns a one class support vector machine [158] from the windows extracted from the training time series, and then assigns a score of +1 or -1 to each test time series based on the prediction of the one class SVM. The anomaly score for the test time series is equal to the inverse of the average score of all its windows. $WINC_{kNN}$ assigns an anomaly score to each window of a test time series equal to the *Euclidean* distance between the window and its k^{th} nearest neighbor in the set of windows extracted from the training time series. The anomaly score for the test time series is equal to the average score of all its windows. Both variants of $WINC$ estimate the density of w-dimensional windows using the windows extracted from the training time series and then determine if the windows extracted from a test time series lie in the dense regions or not.

We evaluated two variants of $WIND$: $WIND_{kNN}$ and $tSTIDE$. $WIND_{kNN}$ is similar to $WINC_{kNN}$ with the only difference being that the score for each window from a test sequence is a likelihood score (inverse of anomaly score), and is equal to the similarity between the window and its k^{th} nearest neighbor in the set of windows extracted from the training sequences. The similarity is measured using *weighted Simple Matching Coefficient* ($wSMC$) measure. For $tSTIDE$ (proposed by Forrest et al [59] for symbolic sequences), the likelihood score for each window is equal to the number of times the window occurs in the set of windows extracted from the training sequences divided by the total number of windows extracted from the training sequences. For both variants of $WIND$, the anomaly score for the test time series is equal to the inverse of the average score of all its windows.

3. Predictive Techniques These techniques learn a predictive model from the training time series. Testing involves forecasting the next observation in a test time series, using the predictive model and the test time series observed so far, and comparing the forecasted observation with the actual observation to determine if an anomaly has occurred.

We evaluate three predictive techniques: AR (using Auto Regressive models [63]), SVR (using Support Vector Regression [134]), and $FSAz$ (using Finite State Automata based technique for symbolic sequences [38]). For AR and SVR, the anomaly score for each observation in a test time series is equal to the difference between the forecasted and the actual observation. The anomaly score of the test time series is equal to the average anomaly scores for all of its observations. For $FSAz$, a likelihood score is obtained for each symbol in the discretized test sequence which is equal to the conditional probability of observing the symbol in the training sequences, given the previous few symbols. The anomaly score for the discretized test sequence is equal to the inverse of the average likelihood scores for all of its symbols.

For comparison, we also evaluate an existing segmentation based technique for time series anomaly detection, called BOX [32]. This technique segments the training time series into fixed number of segments (k) and then learns a sequence of discrete states that define the transition between different segment. A Finite State Automaton (FSA) is defined using these states. The FSA is used to predict the anomalous nature of the test time series.

7.3 Data Sets

Table 7.1 summarizes the different data sets for the cross-domain experimental evaluation. We used 19 data sets grouped into 8 categories with distinct characteristics. For each data set we report the different characteristics as discussed in Section 7.1. The last column denotes the cycle length when the normal time series are periodic. For non-periodic time series, the last column denotes the length of a definitive pattern in the time series. For example, for the valve data set (see Figure 7.2(a)), the "bump" between time instances 100 and 400 can be considered as a pattern. For some non-periodic data sets (disk1–3, shape1–2), it was not possible to define a characteristic pattern. Note that we adapted the actual data sets to create suitable evaluation data sets. The general methodology to create the data sets was the following:

For each data collection, a normal database, $\mathbf{N}$, and an anomalous database, $\mathbf{A}$, of time series is created. A training (reference) database, $\mathbf{X}$, is created by randomly sampling a fixed number of time series from $\mathbf{N}$. A test database, $\mathbf{Y}$, is created by randomly sampling m normal time series from $\mathbf{N} - \mathbf{X}$ and n anomalous time series from $\mathbf{A}$. All time series were normalized to have a zero mean and unit standard deviation. The ratio $\frac{n}{m+n}$ determines the "baseline level" of accuracy for the given test database, e.g., if baseline accuracy is 0.50, a "dumb" technique that declares all time-series to be anomalous will perform correctly 50% of the time. Thus a real technique should perform significantly better than the baseline to be considered effective. The different data sets are:

- **Disk Defect Data Set (disk)** [164]. The normal time series corresponds to blade tip clearance measurements obtained from a simulated aircraft engine disk and the anomalous time series correspond to the measurements obtained to a disk with a crack. Different data sets (disk1 – disk3) correspond to different speeds at which the disk is

			$\|\mathbf{Y}\|$						
Name (#)	L	$\|\mathbf{X}\|$	$\|\mathbf{Y_N}\|$	$\|\mathbf{Y_A}\|$	λ	A	P	S	l
disk (3)	64	10	500	50	0.09	s	$\times$	$\surd$	–
motor (4)	1500	10	10	10	0.50	p	$\surd$	$\times$	250
power (1)	672	11	33	8	0.19	s	$\surd$	$\surd$	96
valve (1)	1000	4	4	8	0.67	s	$\times$	$\surd$	300
shape1 (1)	1614	10	10	10	0.50	p	$\times$	$\times$	–
shape2 (1)	1614	30	30	10	0.25	p	$\times$	$\times$	–
l-ecg (4)	2500	250	250	25	0.09	s	$\surd$	$\times$	250
s-ecg (4)	360	500	500	50	0.09	p	$\times$	$\times$	50

Table 7.1: Details of different data sets used in the experiments. # - number of data sets in each group, L - length of sequences, $\mathbf{X}$ - Training database, $\mathbf{Y}$ - Test database, $\mathbf{Y_N}$ - Normal test time series, $\mathbf{Y_A}$ - Anomalous test time series, λ - Baseline Accuracy, A - Anomaly Type (Process - p, Subsequence - s), P - Periodic (Yes - $\surd$, No - $\times$), S - Synchronized (Yes - $\surd$, No - $\times$), l - Cycle/Characteristic Pattern Length.

rotating.

- **Motor Current Data Set (motor)** [99]. The normal time series correspond to the current signals from the normal operation of a induction motor. The anomalous time series correspond to signals obtained from a *faulty* motor. Different data sets (motor1 – motor4) correspond to different types of faults in the motor.

- **Power Usage Data (power)** [99]. The normal time series correspond to weekly time series of power consumption at a research facility in 1997 for weeks with no holidays during the weekdays. The anomalous time series correspond to power consumption during weeks with a holiday during the week.

- **NASA Valve Data (valve)** [99]. The normal time series consists of TEK solenoid current measurements recorded during the normal operation of a *Marrotta* series valves located on a space shuttle. The anomalous time series correspond to the faulty operation of the valve.

- **Shape Data (shape1 and shape2)** [99]. The normal time series correspond to one or more shapes, while the anomalous time series correspond to other shapes. For the **shape1** database, the normal time series correspond to a pencil shape, while the anomalous time series correspond to other similar shapes (e.g., fork). For the **shape2** database, the normal time series correspond to shapes of cups, glasses, and crosses, while the anomalous time series correspond to distinctly dissimilar shapes.

- **Electrocardiogram Data (l-ecg and s-ecg)** [72]. Each data set corresponds to an ECG recording for one subject suffering with a particular heart condition. The ECG recording is segmented into short time series of equal lengths. Each short time series is added to the normal database if it does not contain any annotations of a heart condition[2], and is added to the anomalous database if it contains one or more annotations indicating a heart condition. The **l-ecg** databases contain 10 second long time series. Four such databases (l-ecg1–l-ecg4) were constructed from BIDMC Congestive Heart Failure Database and Long-Term ST Database. The **s-ecg** databases contain 1 second long time series. Four such databases (s-ecg1–s-ecg4) were constructed from European ST-T Database and MIT-BIH Arrhythmia Database.

7.4 Experimental Results

In this section we summarize the results of our extensive experiments to evaluate 10 anomaly detection techniques, discussed in Section 7.2, on 19 different publicly available time series data sets (See Table 7.1). For each technique we tested various parameter combinations. We report the accuracy (defined in Chapter 3) as the performance metric. Note that, a technique performs well on a given data set if its accuracy is significantly greater than the baseline accuracy for that data set as shown in Table 7.1. Thus for the l-ecg data sets, if the accuracy is 50%, the performance is good since its baseline accuracy is 0.09, while on motor data sets, a 50% accuracy is poor since its baseline accuracy is 0.50. For brevity, we show average results for the 8 groups of data sets.

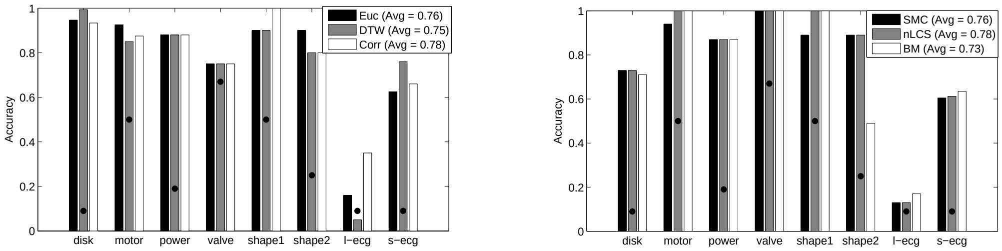
For techniques that operate on discrete sequences, we discretized the time series using the SAX approximation technique.We experimented with different parameter settings for SAX and report results for the best performing combination.

7.4.1 Kernel Based Techniques

We experimented with different distance and similarity measures for the kernel based techniques, $KNNC$ and $KNND$, as well as different values for parameter k. We observed that for both $KNNC$ and $KNND$, the best performance was generally obtained when $k \in (2, 8)$.

---

[2] http://www.physionet.org/physiobank/annotations.shtml



(a) KNNC

(b) KNND

Figure 7.3: Results for Kernel Based Techniques. "•" indicates the baseline accuracy for each data set.

The results for $KNNC$ using different distance measures are summarized in Figure 7.3(a). Overall, $KNNC$ performs well for most techniques, but the performance varies somewhat with the choice of the distance measure. The cross-correlation measure is consistently better or similar in performance to the Euclidean measure for most data sets, especially when the normal time series are not synchronized. The reason is that Euclidean measure cannot capture phase misalignments present among the normal time series while cross-correlation can. Although, on an average, cross correlation outperforms DTW, both these measures show complementarity. For periodic data sets (e.g., motor, l-ecg), cross correlation is better than DTW and for non-periodic data sets (e.g., disk, s-ecg), DTW tends to be better than cross-correlation. DTW addresses the issue of non-linear alignment of time series while cross-correlation addresses the issue of phase misalignments. The reason why DTW is poor for periodic time series is that when a normal periodic time series is compared with an anomalous periodic time series using DTW, the anomalous portion is ignored by DTW (to obtain a better "match"), and hence the distance between the two time series is almost equal to the distance between two normal periodic time series. The reason why cross-correlation performs poorly on non-periodic time series data is that for an anomalous time series cross-correlation finds the best "phase shifted" version of a normal time series. Even if the anomalous time series are generated from a different process, it might be similar to some phase shifted version of a normal time series, and hence the anomalous time series would appear normal.

The results for $KNND$ are summarized in Figure 7.3(b). For the *BitMap* (BM)

measure, we used different values of the bitmap resolution and reported best results for resolution 2. The similarity measure $nLCS$ performs best; while SMC and BM are also comparable in performance. Interestingly, the SMC measure always performs better than the weighted SMC, and hence is omitted from Figure 7.3(b).

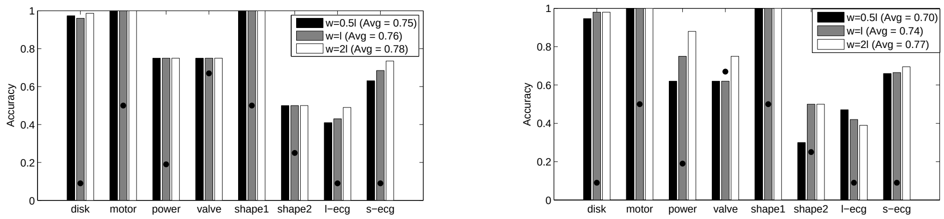
Comparing results in Figures 7.3(a) and 7.3(b), we observe that $KNNC$ with cross-correlation and $KNND$ with $nLCS$ perform consistently well. In some cases, approximating the time series by discretization results in loss of information, which negatively impacts the performance of $KNND$ (disk), though in certain cases, when the data is noisy, the approximation removes the noise in the time series and hence results in better performance (valve).

$KNNC$ and $KNND$ are linear in the size of training and testing data sets ($|X|$ and $|Y|$). $KNNC$ with *Euclidean* distance measure and $KNND$ with SMC similarity measure are linear in the length of time series ($O(L)$). $KNNC$ with DTW and $KNND$ with $nLCS$ are quadratic $O(L^2)$, though the complexity can be improved by using faster heuristics [104, 28]. $KNNC$ with *cross correlation* is $O(L \log L)$. $KNND$ with the Bitmap measure is linear in the length of time series, but is $O(b^3)$ where b is the length of the window used for computing the similarity.

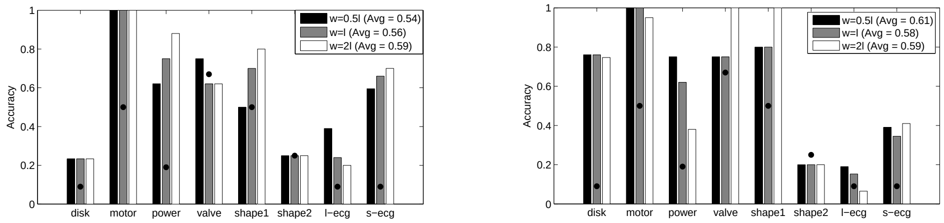
7.4.2 Window Based Techniques

For window based techniques, we experimented with varying window lengths. For data sets with long time series ($L \geq 1000$) we used a sliding step of 50 when extracting windows from the training and test time series. The results for the four window based techniques are summarized in Figures 7.4(a) – 7.4(d).

The window length w is a critical parameter for these techniques, e.g., in $WINC_{kNN}$ if $w = 2$, all windows are likely to find almost identical nearest neighbors, and hence no difference between normal and anomalous test time series can be detected. If w is very large, nearest neighbors for all windows are likely to be far, and hence the normal and anomalous test time series cannot be differentiated. We investigated different reasonable settings based on the length of a cycle or "characteristic pattern" in the normal time series (See Table 7.1). We experimented with different multiples of the cycle length, as well as a fixed short length of 10. Since the results for $w = 10$ were never competitive, those results are not shown here.



(a) $WINC_{SVM}$ (b) $WINC_{KNN}$



(c) $WIND_{KNN}$ (d) tSTIDE

Figure 7.4: Results for Window Based Techniques. "●" indicates the baseline accuracy for each data set.

$WINC_{SVM}$ performs consistently better than other window based techniques on most of the data sets. For $WINC_{SVM}$ we used a Radial Basis Function (RBF) kernel, with kernel width $\gamma = 0.1$. In our experiments the results did not vary significantly with varying γ. An important parameter for one class SVM is ν [158], which is an upper bound on the fraction of training instances that are treated as the "other" class. The results show that for data sets with noisy training time series, a higher value of $\nu \approx 0.1$ gave best results, while for data sets with less noisy training time series, a lower value of $\nu \approx 0.001$ gave best results. Thus ν can be set according to the nature of the underlying time series. The performance of $WINC_{SVM}$ does not depend on the window size w for most of the data sets.

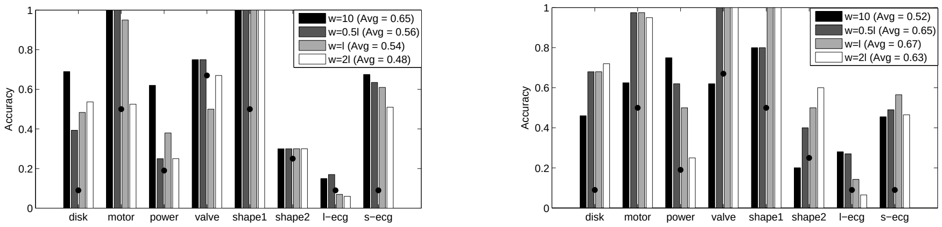
We experimentally verified that the performance of $WINC_{kNN}$ gives good results for values of $nn \in [5, 30]$. For $WIND_{kNN}$, since continuous values are mapped to a limited number of symbols, and hence more windows from test sequences find exact matching windows in training sequences, and hence the performance of $WIND_{kNN}$ is good only when $nn \approx 100$.

Results indicate that the techniques for continuous data ($WINC_{SVM}$ and $WINC_{KNN}$)

are generally better than those for discretized data ($WIND_{KNN}$ and $tSTIDE$), indicating that the loss of information due to discretization has a negative effect on the performance. Another interesting observation is that the performance of window based techniques is impacted more significantly due to the symbolic approximation than the performance of kernel based techniques. The reason for this is that when the time series are discretized using few symbols, any short window from a test sequence, in the discretized form, is likely to find exact matching windows in the training sequences. For certain data sets, performance of $WINC_{KNN}$ depends on the choice of w, e.g., for power data set in Figure 7.4(b), the performance improves when the window length increases, while for l-ecg data sets, the performance deteriorates when the window length increases. In general, for $WINC_{KNN}$, window length has more impact on the performance for periodic data sets and not for non-periodic data sets.

Performance of $WINC_{SVM}$ and $KNNC$ is comparable on average. $WINC_{SVM}$ is generally better suited when the time series are long (l-ecg), because for $KNNC$, comparing two long time series suffers from the *curse of dimensionality* while $WINC_{SVM}$ breaks the time series into short windows, and hence does not get significantly affected by the length of the time series. One data set where $KNNC$ is significantly better than $WINC_{SVM}$ is shape2. In this data set, the normal time series belong to three different clusters. $KNNC$ is well-suited to utilize this information because the normal time series form tight neighborhoods within their clusters. $WINC_{SVM}$ does not utilize this information and hence performs poorly.

The training time for $WINC_{SVM}$ can at worst be $O(|X|^3L^3)$, but is typically $O(N_sw|X|L)$, if the number of support vectors, N_s, is very few. Testing for $WINC_{SVM}$ is linear in $|Y|$, L, w, and N_s. $WINC_{KNN}$ and $WIND_{KNN}$ do not have a training phase. The testing phase is linear in $|X|$, $|Y|$, and w, but the is quadratic in L $(= O(|X||Y|L^2w))$. The testing time for $WINC_{KNN}$ and $WIND_{KNN}$ can be improved by using indexing strategies that have been proposed for discord detection [101, 100, 62, 27]. The training time for $tSTIDE$ is linear in $|X|$, L, and the window length w, while testing is also linear in $|Y|$, $|L|$ and w.



(a) SVR (b) FSA-z

Figure 7.5: Results for Predictive Techniques. "•" indicates the baseline accuracy for each data set.

7.4.3 Predictive Techniques

For the predictive techniques the critical parameter is the size of the history used to obtain the forecast at a given time instance. For notational simplicity, we will consider the size of history equivalent to the window length parameter for window based techniques and denote it with w. The results for the predictive techniques are summarized in Figures 7.5. The results of AR were poor across all data sets, and hence are not shown.

SVR requires additional parameters for the support vector regression, such as the tradeoff between training error and margin (C), the width of tube for regression (ϵ), choice of kernel, and kernel parameters. We used default values for C. We experimented with different kernels and reported best performance with the RBF kernel with $\gamma = 1.00$. The optimal value for ϵ is related to the presence of noise in the training time series. If the normal time series are noisy (e.g., l-ecg), a higher value of $\epsilon \approx 0.4$ gave best results, while if the normal time series is less noisy (e.g., motor), a lower value of $\epsilon \approx 0.05$ gave best results. The reason being that a small value of ϵ results in a tight regression tube and if a normal time series is noisy, several observations may be considered anomalous, resulting in a high false positive rate. Whereas, a large value of ϵ results in a relaxed regression tube, which may result in several truly anomalous observations to be missed.

SVR technique performs better for low value of w. Results for lower values of $w < 10$ also gave similar performance as $w = 10$. The reason being that for longer history, SVR cannot learn an accurate regression model. The results for data with shorter time series, such as s-ecg, are promising, though a more thorough investigation of the parameter space is required for more accurate insights.

The performance of $FSAz$ and SVR exhibit complementarity. For the longer data sets (e.g., l-ecg and valve), $FSAz$ outperforms SVR, while for shorter data sets (e.g., s-ecg), SVR is better. The reason is that the discretization reduces the noise in the data, and hence $FSAz$ is able to learn a more accurate predictive model than SVR.

In general, SVR and $FSAz$ do not perform as well on most data sets as $KNNC$ and $WINC_{SVM}$. This indicates that for continuous time series as well as the discretized sequences, learning an accurate predictive model for anomaly detection is challenging.

The training time for

$$SVR$$

can at worst be $O(|X|^3L^3)$, but is typically $O(N_s w|X|L)$, if the number of support vectors, N_s, is very few. Testing for

$$SVR$$

is linear in $|Y|$, L, w, and N_s. For $FSAz$, both training and testing phases are linear in $|X|$ (and $|Y|$), L, and w.

7.4.4 Segmentation Based Technique - BOX

For the BOX technique, we experimented with different number of boxes, k.

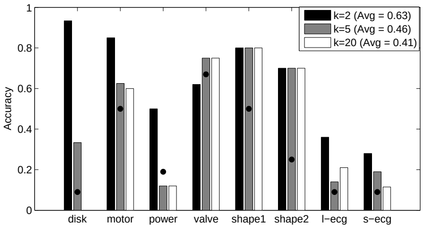
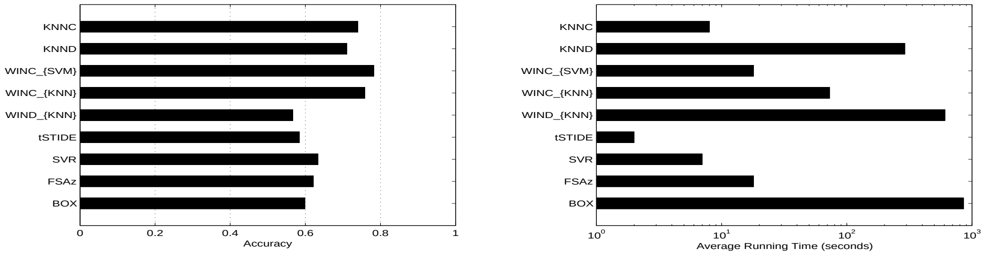


Figure 7.6: Results for BOX Technique. "•" indicates the baseline accuracy for each data set.

The results are summarized in Figure 7.4.4. The performance of BOX is highly sensitive to the parameter k and gives best performance for $k = 2$ or 3. BOX is outperformed by $KNNC$ and $WINC_{SVM}$ for most data sets. The original paper for BOX



(a) Accuracies

(b) Running times in seconds (X-axis in log scale)

Figure 7.7: Average performance for all techniques for best performing parameter setting.

[32], evaluated the technique on the valve data set, but we found that other techniques ($KNND$ and $FSAz$) outperformed BOX on this data set. Also, the performance of BOX is best for $k = 20$ on valve data set, which was also reported as the best value by Chan et al, while for most of the other data sets, the best performance was obtained for $k = 2$. For the BOX technique, training phase is linear in $|X|$ but has cubic complexity in L, $(= O(L^3))$. The testing phase is linear in $|Y|$ and L.

7.5 Discussions and Conclusions

The average accuracies for all techniques across all 19 data sets are shown in Figure 7.7(a). For each technique we also show the average running time (training + testing), across all data sets in Figure 7.7(b). The results reveal several interesting insights into the performance of the different techniques. At a high level, our results indicate that none of the techniques are superior to others across all data sets, but have certain characteristics that make them effective for certain types of data sets, and ineffective for certain others.

Here we summarize some high level conclusions:

- Techniques that operate on the continuous time series are generally superior to techniques that operate on discrete sequences. Moreover, techniques for continuous time series are more robust to parameters such as distance measure ($KNNC$ vs. $KNND$) or window length ($WINC_{KNN}$ vs. $WIND_{KNN}$).

- For the window based techniques, using one class SVMs (with RBF kernel) to estimate the density of the windows is more effective than using a nearest neighbor based approach ($WINC_{SVM}$ vs. $WINC_{KNN}$).

- Overall, kernel and window based techniques, which are model independent, tend to outperform predictive and segmentation based techniques, that try to build a model for the time series data. This seems to indicate that building an AR or SVR model for time series data is challenging. This is interesting, given that for symbolic sequences, it has been shown that similar predictive techniques and kernel based techniques have comparable performance [38].

- For kernel based techniques, the choice of distance or similarity measure is critical, and for window based techniques, the choice of window length is critical. Kernel based techniques are faster than window based techniques, though indexing techniques, that were originally proposed to improve the time complexity of discord detection techniques [101, 100, 62, 27], can be employed for $WINC_{KNN}$ and $WIND_{KNN}$ to make them faster. If online anomaly detection is desired, techniques such as $KNNC$ and $KNND$, which require the knowledge of entire test time series are not suitable, while window based and predictive techniques can be adapted to operate in an online setting.

- For periodic time series data, window based techniques perform superior to other techniques, e.g., $WINC_{SVM}$ for l-ecg. The reason being that if the training time series are periodic, they can be represented using a small set of windows which form dense regions in the w dimensional space. Thus one class SVMs (as well as nearest neighbor based density estimation), can learn a tight boundary around the dense regions, and can differentiate well between the windows from a normal test time series and those from an anomalous time series. On the other hand, for non-periodic time series data, a larger set of windows is required to represent the training time series, and hence the windows form sparse regions in the w dimensional space. Thus the decision surface learnt by the one class SVMs is not discriminatory enough. This results in relatively poor performance for $WINC_{SVM}$ compared to other techniques on non-periodic time series data, e.g., shape2.

- If the time series data contains process anomalies, kernel based techniques give best performance, e.g., $KNND$ for shape2, while the performance is poor if the time series data contains subsequence anomalies, e.g., $KNNC$ for l-ecg. The reason is that the kernel based techniques assume that the anomalous test time series are significantly different from the normal time series, which is true for data with process anomalies, but not for data with subsequence anomalies. For the latter type of data, window based and predictive techniques are better suited since they analyze windows within the time series and are more likely to detect the anomalous subsequences.

- We learnt several relationships between the nature of the normal and anomalous data and the performance of different techniques. For example, $KNNC$ with DTW measure is suited for non-periodic time series while $WINC_{SVM}$ is more suited for periodic time series. $WINC_{SVM}$ performs poorly for data sets in which the normal time series belong to multiple modes (e.g., shape2), while $KNNC$ and $KNND$ are better suited to handle such data sets.

The experimental evaluation indicates that the performance of each technique on a data set relies on the parameter settings, and in many cases we observe that different settings are optimal for different types of data sets. While we investigated many combinations of parameters, exhaustive testing could not be done due to the large parameter space. The ultimate aim of this study is to be able to choose the best technique for a given data set. Our study is a step in this direction. For some techniques such as $WINC_{SVM}$ we have discussed the relationship between the parameters and the nature of data. For a better understanding of the problem, a richer collection of data sets, as well as deeper understanding of the algorithms and the generative processes for the time series data, is required. Most of the periodic time series data sets used in this chapter have a single dominant frequency in the time series. For such data, certain techniques like $WINC_{SVM}$ are well suited. It needs to be investigated in future if the performance of different techniques varies when the time series are aggregates of multiple frequencies.

While we have shown how anomalies in time series data can be detected in a number of ways, this is not an exhaustive set of possibilities. Time series data has also been widely studied in contexts other than anomaly detection, in research communities such as *signal processing* and *statistics*, though not directly in the context of anomaly detection problem. Some of this research has been, to a limited extent, and can be, in future, adapted to solve the anomaly detection problem. For example, signal processing techniques such as *fourier analysis* can be used to transform time series data into frequency domain, and perform anomaly detection in the frequency domain. Chen and Zhan [193] employ wavelets to extract features from time series data and perform anomaly detection on the extracted features. Similarly, several statistical models time series data have been widely researched in the statistics community. In this chapter, we have used two models, viz, auto-regression and support vector regression, for anomaly detection and shown that such models were not able to capture the normal behavior of the time series data.. In future, other sophisticated models, such as *Gaussian Process Regression*, etc., can be adapted in a similar fashion.

Chapter 8

Anomaly Detection for Multivariate Time Series Data

Multivariate time series data is relevant in many application domains in which data is naturally collected as multivariate time series. For example, a single aircraft flight generates a multivariate time series where each individual time series corresponds to data coming from a single sensor or switch on the aircraft [165]. Similarly, in the case of network intrusion detection, a daily network log is a multivariate time series such that each variable measures certain aspect of the time series [91].

Anomaly detection for multivariate time series data is distinct from traditional anomaly detection techniques for multivariate data as well as for univariate time series data. This is because while the former only analyze the multivariate aspect of the data, the latter only analyze the sequence aspect for individual variables. Often, the anomaly in a multivariate time series can be detected only by analyzing sequence of all (or a subset of) variables.

For example, Figure 8.1 shows an example of an anomaly in a multivariate time series. Figure 8.1(a) shows an artificially generated time series with 5 attributes. Figure 8.1(b) shows a similar multivariate time series with artificially induced anomalies between time points 40 and 50. It is not possible to differentiate between the normal and anomalous time series by analyzing the five univariate time series individually using the techniques available for univariate sequence data, or by analyzing the five dimensional observations independently, using traditional anomaly detection techniques for multivariate (non-sequence) data. To detect such anomalies in multivariate sequences, multivariate aspect as well as the sequence aspect needs to be simultaneously modeled.

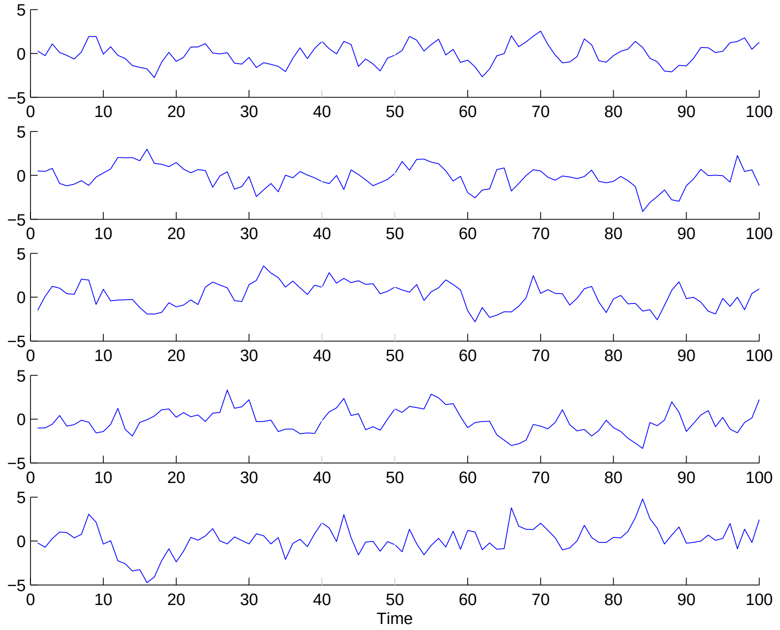
In this chapter we present a novel anomaly detection technique, WIN_{SS} to solve the semi-supervised anomaly detection problem (See Chapter 3) for multivariate time series data. The proposed technique is a window based technique that accounts for both multivariate as well as sequence aspect of the data while detecting anomalies. The key underlying idea is to reduce a multivariate time series into a univariate time series by exploring the change in the correlation structure of the time series using subspace monitoring.

The rest of this chapter is organized as follows. We discuss the concept of subspace monitoring for multivariate time series in Section 8.1. We present a window based anomaly detection technique for multivariate time series using subspace monitoring, WIN_{SS}, in Section 8.2. The multivariate time series data used for evaluation is described in Section 8.3. The experimental evaluation of the proposed technique is provided in Section 8.4.

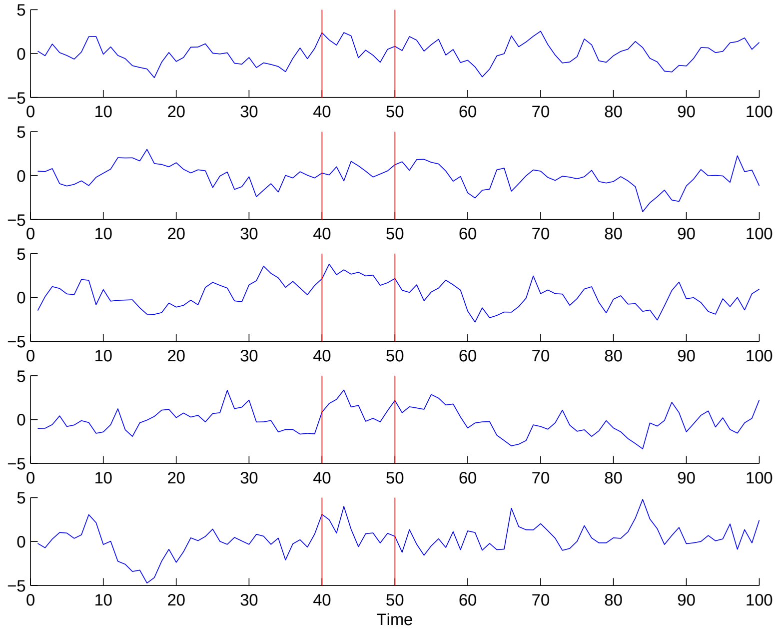
8.1 Subspace Monitoring for Multivariate Time Series

A vast literature exists for subspace monitoring for damage detection and other areas falling under the broad purview of *statistical process control* [14, 98]. The original concept of comparing two subspaces using the angles between their principal components was proposed by Jordan et al [95] in the 19^{th} century and was explained as *canonical correlation* by Hotelling [88].

Let $A \in \Re^{m \times p}$ and $B \in \Re^{m \times q}$ be the matrices of orthogonal basis vectors for two subspaces, S_A and S_B, respectively, of an original $m \times m$ vector space. Let $q \leq p$. The



(a) Normal Time Series



(b) Anomalous Time Series (Anomalies injected from time point 40 to 50).

Figure 8.1: Anomaly in a Multivariate Time Series

q principal angles between the two subspaces, $\theta_k, 1 \leq k \leq q$, are defined as:

$$\begin{aligned}
\cos\theta_1 &= \max_{\substack{x\in\Re^p \\ y\in\Re^q}} \frac{|x^T A^T B y|}{||Ax||_2||By||_2} = \frac{|x_1^T A^T B y_1|}{||Ax_1||_2||By_1||_2} && (8.1) \\
\cos\theta_k &= \max_{\substack{x\in\Re^p \\ y\in\Re^q}} \frac{|x^T A^T B y|}{||Ax||_2||By||_2} = \frac{|x_k^T A^T B y_k|}{||Ax_k||_2||By_k||_2} && (8.2) \\
& \qquad subject\ to\ x_i^T A^T A x = 0\ and\ y_i^T B^T B y = 0,\ for\ i = 1, 2, \ldots, k-1
\end{aligned}$$

$Ax_k \in range(A)$ and $By_k \in range(B)$ denote the k^{th} principal directions. Note that the principal angles satisfy $0 \leq \theta_1 \leq \ldots \leq \theta_q \leq \frac{\pi}{2}$. The ordered list of principal angles between subspaces S_A and S_B is also denoted as:

$$(\theta_1, \theta_2, \ldots, \theta_q) = [A \lessdot B] \tag{8.3}$$

Golub et al [73] have shown using *Lagrange Multiplier* formulation that the principal angles and the directions between a pair of subspaces can be solved as the following *generalized eigenvalue problem*:

$$\begin{pmatrix} 0 & A^T B \\ B^T A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} A^T A & 0 \\ 0 & B^T B \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \lambda \tag{8.4}$$

$$subject\ to\ x^T A^T A x = 1\ and\ y^T B^T B y = 1$$

Let the eigenvalues obtained from the problem in (8.4) be $\lambda_1, \lambda_2, \ldots, \lambda_{p+q}$ and

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{p+q}.$$

Then it has been shown that:

$$\cos\theta_1 = \lambda_1, \ldots, \cos\theta_q = \lambda_q \tag{8.5}$$

Further, the squared cosines of the principal angles can also be shown to be equal to the eigenvalues of the matrix $(AA^T)^{-1}AB^T(BB^T)^{-1}BA^T$.

The corresponding principal directions Ax_k and By_k are obtained using the k^{th} eigenvector corresponding to λ_k. Though the generalized eigenvalue approach can be used to compute the principal angles, numerical approaches using Singular Value Decomposition (SVD) have been proposed to obtain the principal angles [73].

The comparison of the subspaces has been used for indirect comparison of multivariate time series by comparing their generative models, such as vector AR models [47], ARMA models [23], and linear dynamical systems [14].

Measuring Change Between a Pair of Subspaces

The principal angles between a pair of subspaces can be computed using (8.4) and (8.5). For the anomaly detection technique proposed in this chapter, it is sufficient to measure the change between two subspaces, S_A and S_B, which is defined as the following [98]: *Change in subspace when S_A changes to subspace B is equal to the maximum distance between an arbitrary unit vector $\hat{x}$ in S_B and the subspace S_A.* This change δ_{AB} is given as:

$$\delta_{AB} = \sqrt{1 - \lambda_{min}} \tag{8.6}$$

where λ_{min} is the minimum eigen value of $B^T(AA^T)B$. A and B are the m-dimensional basis vectors of subspaces A and B, respectively. For proof see [98].

8.2 Anomaly Detection for Multivariate Time Series Using Subspace Monitoring

We propose a novel anomaly detection techniques, WIN_{SS} which consists of two steps. In the first step each training and test multivariate time series is converted into a univariate time series. In the second step, an existing anomaly detection technique for univariate time series (see Chapter 7) is applied on the converted univariate data to detect anomalies.

8.2.1 Converting A Multivariate Time Series to Univariate Time Series

Let $S \in \Re^{|S| \times m}$ be a multivariate time series of length $|T$ and consisting of m variables.

Let a w length window of S starting at time t be denoted as $W_t = S[t]S[t+1]\ldots S[t+w-1]$. Thus we can extract $T - w + 1$ such windows from S. Consider two successive windows $W_t, W_{t+1} \in \Re^{w \times m}$. Let V_t denote the subspace spanned by the top *few*[1] principal components of W_t. Similarly V_{t+1} denote the subspace spanned by the top *few* principal components of W_{t+1}. Note that W_t and W_{t+1} can have different number of basis vectors. The change between the two successive subspaces, $\delta_{t,t+1}$ can be defined

---

[1] Capturing α% of the total variance.

				$\|\mathbf{Y}\|$		
Name	L	d	$\|\mathbf{X}\|$	$\|\mathbf{Y_N}\|$	$\|\mathbf{Y_A}\|$	λ
CMAPSS	4122	30	25	16	4	0.20
EEG	256	64	100	200	50	0.20

Table 8.1: Details of different multivariate time series data sets used in the experiments. # - number of data sets in each group, L - length of sequences, d - number of variables, $\mathbf{X}$ - Training database, $\mathbf{Y}$ - Test database, $\mathbf{Y_N}$ - Normal test time series, $\mathbf{Y_A}$ - Anomalous test time series, λ - Baseline Accuracy.

using (8.6) as:

$$\delta_{t,t+1} = \sqrt{1 - \lambda_{min}} \tag{8.7}$$

where λ_{min} is the minimum eigenvalue of the matrix $V_t^T V_{t-1} V_{t-1}^T V_t$. Thus the multivariate time series S can be transformed into a univariate time series $\delta_{1,2}\delta_{2,3}\ldots$.

8.2.2 Detecting Anomalies in Converted Data

The transformed univariate time series captures the dynamics of the subspace structure of a multivariate time series, such that the normal time series are expected to follow similar dynamics, while anomalous time series are different. We use $WINC_{SVM}$ to detect anomalies in the transformed univariate test data set.

8.3 Data Sets Used

To evaluate the performance of the proposed WIN_{SS} technique, we use artifically generated time series data and two publicly available multivariate time series data sets from the domains of aircraft safety and neurosciences. The details of the public data sets are summarized in Table 8.1.

8.3.1 Artificial Data

To generate the normal time series (Figure 8.1), we first generate four independently generated univariate time series using a first order autoregressive model of length 100. Let the four time series be denoted by $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \mathbf{x_4}$. Let $\mathbf{\Delta_n} = \mathbf{0}$ denote a 100 length vector of all zeros. We use the following mixing matrix $\mathbf{M}$ to generate a 5 dimensional multivariate time series:

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 1 \end{pmatrix}$$

The normal multivariate time series $\mathbf{N}$ is generated as follows:

$$\mathbf{N} = [\mathbf{x_1}\ \mathbf{x_2}\ \mathbf{x_3}\ \mathbf{x_4}\ \boldsymbol{\Delta}_\mathbf{n}]\,\mathbf{M}$$

To generate the anomalous time series (Figure 8.1) we use $\boldsymbol{\Delta}_\mathbf{a}$ as a 100 length vector with all zeros except for $\boldsymbol{\Delta}_\mathbf{a}[\mathbf{41}] = \boldsymbol{\Delta}_\mathbf{a}[\mathbf{42}] = \ldots = \boldsymbol{\Delta}_\mathbf{a}[\mathbf{50}] = 1$. The anomalous time series $\mathbf{A}$ is generated as follows:

$$\mathbf{A} = [\mathbf{x_1}\ \mathbf{x_2}\ \mathbf{x_3}\ \mathbf{x_4}\ \boldsymbol{\Delta}_\mathbf{a}]\,\mathbf{M}$$

8.3.2 CMAPSS Data

The first data set is an aircraft simulation data set, available from the *NASA Dash-Link Archive* [164]. This data set was generated with the *Commercial Modular Aero-Propulsion System Simulation* (CMAPSS) simulator which is a tool for the simulation of realistic large commercial turbofan engine data. The flights are full flight recordings sampled at 1 Hz and consist of 30 engine and flight condition parameters. Flights are of different lengths. The parameters for each flight are the flight conditions, health indicators, measurement temperatures and pressure measurements.

Each normal flight corresponds to a combination of a series of flight conditions, arranged to cover a typical ascent from sea level to 35K ft and descent back down to sea level. For each anomalous flight, a fault is injected during the flight at a given time. For two anomalous flights, the fault is intermittent, i.e., the flight resumes normal operation after fault, while for the other two flights, the fault is persistent, i.e., the flight remains in faulty state after the first occurrence of the fault.

8.3.3 EEG Data

The second data set consists of *Electroencephalogram* (EEG) signals captured from a set of individuals as a response to a visual stimuli [16]. Each time series consists of 64 variables which correspond to the readings from 64 electrodes placed on the scalp of the individuals.

The normal multivariate time series correspond to control or healthy individuals, while the anomalous time series correspond to individuals under alcoholic influence.

8.4 Experimental Results

In this section we provide two sets of experimental results for the proposed WIN_{SS} technique. First set of results are on the artificially generated multivariate time series shown in Figure 8.1. The second set of results are on the publicly available data sets. For comparison, we also provide results for following two techniques:

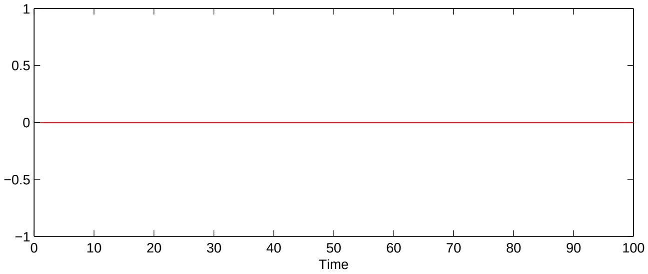
1. ***k*-nearest neighbor technique for multivariate data** (kNN). This technique ignores the sequential aspect of the data. The training and test data set is treated as sets of multivariate observations. Each test observation is assigned an anomaly score as distance to its k^{th} nearest neighbor in the training set. The anomaly score of a test time series is equal to the average anomaly score of the individual observations.

2. **Window based technique for univariate time series** ($WINC_{SVM}$). This technique aignores the multivariate aspect of the data. A univariate time series data set (training and testing) is constructed for each variable. Each univariate test time series is assigned an anomaly score using the $WINC_{SVM}$ technique (See Chapter 7). The anomaly score of a multivariate test time series is equal to the average anomaly score of the individual univariate time series.

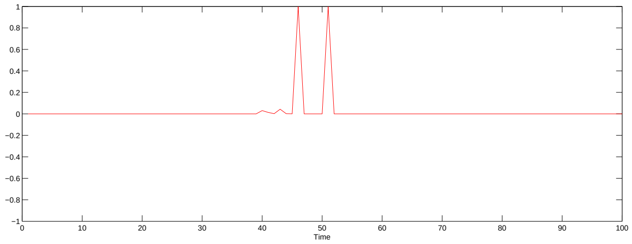
For all three techniques we experimented with different parameter settings are report the results for the best setting here. For WIN_{SS} the different parameters are: window length, w and the percentage of total variance captured by a subspace, α. Experimental results indicate that WIN_{SS} performs best for windows size $w \approx 10$. The performance is not sensitive to the choice of α for $0.85 \leq \alpha \leq 0.95$. For $\alpha > 0.95$, the successive subspaces for normal windows become unstable and hence result in a high false alarm rate. For $\alpha < 0.85$, the successive subspaces for anomalous windows do not differ significantly and hence result in a low detection rate.

8.4.1 Illustration on Artificial Data Set

To illustrate how WIN_{SS} detects anomalies in a multivariate time series, we use the artificially generated multivariate time series shown in Figure 8.1. The univariate time series obtained by applying the WIN_{SS} transformation on the normal and anomalous time series are shown in Figure 8.2.



(a) Transformed Normal Time Series



(b) Transformed Anomalous Time Series

Figure 8.2: Univariate Time Series Obtained using WIN_{SS} on Artificial Data Sets

Figure 8.2 shows that WIN_{SS} assigns a high value to start and end of the anomalous region for the anomalous time series. For the normal time series, the transformed univariate time series is constant since there is no change in the underlying subspace, while for the anomalous time series, the subspace changes abruptly when an anomaly is injected as well as when the anomalous region ends.

8.4.2 Comparing With State of Art

We compared the performance of WIN_{SS} against the two adapted state of the art techniques, kNN and $WINC_{SVM}$ on the public data sets using the two evaluation metrics, *Accuracy* and AUC. The results for accuracy (or precision on anomaly class) and AUC are shown in Table 8.2.

(a) Accuracy

	WIN_{SS}	kNN	$WINC_{SVM}$
CMAPSS	**0.75**	0.50	0.50
EEG	**0.66**	0.17	0.43

(b) AUC

	WIN_{SS}	kNN	$WINC_{SVM}$
CMAPSS	**0.92**	0.81	0.81
EEG	**0.87**	0.72	0.51

Table 8.2: Results on public multivariate time series data.

The results show that WIN_{SS} is more effective than kNN and $WINC_{SVM}$ in detecting anomalies in multivariate time series data. For the CMAPSS data, WIN_{SS} detects 3 out of 4 anomalies with top anomaly scores while the other two techniques detect only 2 out of 4. All the three techniques are able to detect those flights as anomalous for which had *persistent* faults. Only WIN_{SS} was able to detect a fight with *intermittent* fault as anomalous. For EEG data, the performance of kNN is worse than random, since the anomaly is apparent only as a sequence. The performance of $WINC_{SVM}$ is better on EEG, but is still outperformed by WIN_{SS}.

8.5 Conclusions and Future Work

In this chapter we proposed a novel anomaly detection technique to detect anomalies in multivariate time series data. The proposed technique models both sequential as well as multivariate aspect of the data and hence can detect anomalies that cannot be detected by analyzing only one of the two aspects.

We use the concept of subspace monitoring to capture the dynamics of a multivariate time series. While similar idea has been applied for change detection in such data [14], we use the concept to detect anomalies under the general assumption that the dynamics for normal time series is similar to each other and is significantly different for anomalous time series.

After converting a multivariate time series into univariate time series, any anomaly detection technique discussed in Chapter 7 can be applied to detect anomalies. We provide results when using $WINC_{SVM}$. Other techniques also gave similar results.

A critical aspect of the proposed technique is the computation of eigen vectors for the moving window which could become a computational bottleneck, especially when the dimensionality and length of the time series becomes large. A potential solution is to update the eigen vectors for sucessive windows incrementally using techniques such as *Matrix Perturbation* [167] to avoid computation of the eigen vectors for every window, and is suggested as a future direction of research.

Chapter 9

Conclusions

The existing research on anomaly detection for sequences is limited to developing techniques that are suitable for specific application domains. In this thesis we advance the research to the level, where we not only propose new and better techniques but also understand the performance of the techniques across different domains. Each application domain possesses unique characteristics in terms of the data on which the anomaly detection techniques operate. Our study reveals that the performance of the anomaly detection techniques is closely tied to the nature of the underlying data, and hence the techniques exhibit varying performance across application domains. The results of our experimental studies for symbolic sequences and time series data can help a domain scientist find the technique which is best suited for an application domain,

The question that arises from experimental studies is that why certain techniques that perform well on data from one domain, do not perform as well on data from another domain. We answer this question by studying the different characteristics of sequence data and connecting the performance of the anomaly detection techniques to these data characteristics. The RBA framework is a simple, yet powerful tool to analyze and characterize data and understand the performance of different anomaly detection techniques.

An additional use of RBA framework is that the dimensions, into which the original data is mapped, can be utilized as features that can be used to develop better anomaly detection techniques. Since each dimension in RBA is actually utilized by one or more anomaly detection techniques, the new technique, which uses the RBA features, is like a *meta* anomaly detection technique, which combines the strengths of different existing techniques. But since RBA uses a small canonical set of dimensions (separability statistics), one can arguably achieve better than the existing best performance on a given data sets without having to run the several existing techniques.

In future, RBA can be extended in multiple ways. One is to extend RBA to other types of data, such as time series. For this, one needs to devise a set of separability statistics that can be used to characterize time series data sets. The second way is to enhance the set of separability statistics for a given type of data, to achieve better visualization as well as better features for anomaly detection. The third way is to improve the RBA feature based anomaly detection techniques by finding the best possible, linear or non-linear, combination of the RBA features which can distinguish between the normal and anomalous data .

In this thesis, we propose several novel anomaly detection techniques for symbolic sequences, univariate time series data, and multivariate time series data. All of our techniques share a common theme of window based analysis. Window based techniques are effective for anomaly detection in sequences due to two reasons. First, by sliding a window across a sequence, these techniques account for the sequential nature for the data. Second, by analyzing a subsequence of observations as a single unit of analysis, one can detect collective anomalies which cannot be detected by analyzing an individual observation independently. But window based techniques have an issue of redundancy, since successive windows, often, are highly similar to each other and do not provide any additional information. Removing this redundancy can speed up the performance of the technique and also, possibly, improve results. This aspect of redundancy needs to be understood and is suggested as a future direction of research.

In this thesis, we propose novel anomaly detection techniques for multivariate time series data, which is a relatively lesser explored topic. Same principle can be applied for handling multivariate heterogenous data, or multivariate discrete data. The only requirement is to design a measure to compare two successive windows, and is suggested as a future direction of research.

References

[1] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 504–509, New York, NY, USA, 2006. ACM Press.

[2] B. Abraham and A. Chuang. Outlier detection and time series modeling. *Technometrics*, 31(2):241–248, 1989.

[3] A. Agovic, A. Banerjee, A. R. Ganguly, and V. Protopopescu. Anomaly detection in transportation corridors using manifold embedding. In *First International Workshop on Knowledge Discovery from Sensor Data*. ACM Press, 2007.

[4] A. Agresti. *Categorical Data Analysis*. John Wiley & Sons, 2003.

[5] M. Agyemang, K. Barker, and R. Alhajj. A comprehensive survey of numeric and symbolic outlier mining techniques. *Intelligent Data Analysis*, 10(6):521–538, 2006.

[6] E. Aleskerov, B. Freisleben, and B. Rao. Cardwatch: A neural network based database mining system for credit card fraud detection. In *Proceedings of IEEE Computational Intelligence for Financial Engineering*, pages 220–226, 1997.

[7] A. Arning, R. Agrawal, and P. Raghavan. A linear method for deviation detection in large databases. In *Proceedings of 2nd International Conference of Knowledge Discovery and Data Mining*, pages 164–169, 1996.

[8] A. Asuncion and D. J. Newman. UCI machine learning repository. [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, 2007.

[9] Z. Bakar, R. Mohemad, A. Ahmad, and M. Deris. A comparative study for outlier detection techniques in data mining. *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*, pages 1–6, June 2006.

[10] R. Baragona and F. Battaglia. Outliers detection in multivariate time series by independent component analysis. *Neural Computing*, 19(7):1962–1984, 2007.

[11] D. Barbara, J. Couto, S. Jajodia, and N. Wu. Detecting novel network intrusions using bayes estimators. In *Proceedings of the First SIAM International Conference on Data Mining*, 2001.

[12] D. Barbará, Y. Li, and J. Couto. COOLCAT: an entropy-based algorithm for categorical clustering. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 582–589, New York, NY, USA, 2002. ACM.

[13] V. Barnett and T. Lewis. *Outliers in statistical data*. John Wiley and sons, 1994.

[14] M. Basseville, M. Abdelghani, and A. Benveniste. Subspace-based fault detection algorithms for vibration monitoring. *Automatica*, 36:101–109, 2000.

[15] A. Bateman, E. Birney, R. Durbin, S. R. Eddy, K. L. Howe, and E. L. Sonnhammer. The pfam protein families database. *Nucleic Acids Res.*, 28:263–266, 2000.

[16] S. D. Bay, D. F. Kibler, M. J. Pazzani, and P. Smyth. The UCI KDD archive of large data sets for data mining research and experimentation. *SIGKDD Explorations*, 2(2):81–85, 2000.

[17] R. J. Beckman and R. D. Cook. Outlier...s. *Technometrics*, 25(2):119–149, 1983.

[18] F. Bendix, R. Kosara, and H. Hauser. Parallel sets: Visual analysis of categorical data. In *IEEE Symposium on Information Visualization*, page 18, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

[19] D. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *AAAI Workshop on Knowledge Discovery in Databases*, pages 229–248, 1994.

[20] A. Beygelzimer, C.-S. Perng, and S. Ma. Fast ordering of large categorical datasets for better visualization. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–244. ACM, 2001.

[21] J. Blasius and M. Greenacre. *Visualization of Categorical Data*. Academic Press, 1998.

[22] R. Blender, K. Fraedrich, and F. Lunkeit. Identification of cyclone-track regimes in the north atlantic. *Quarterly Journal of the Royal Meteorological Society*, 123(539):727–741, 1997.

[23] J. Boets, K. D. Cock, M. Espinoza, and B. D. Moor. Clustering time series, subspace identification and cepstral distances. *Communications in Information Systems*, 5(1):69–96, 2005.

[24] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In *SDM 2008: Proceedings of the eighth SIAM International Conference on Data Mining*, pages 243–254, 2008.

[25] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Optics-of: Identifying local outliers. In *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 262–270. Springer-Verlag, 1999.

[26] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *Proceedings of 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104. ACM Press, 2000.

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

[28] S. Budalakoti, A. Srivastava, R. Akella, and E. Turkov. Anomaly detection in large sets of high-dimensional symbol sequences. Technical Report NASA TM-2006-214553, NASA Ames Research Center, 2006.

[29] S. Budalakoti, A. Srivastava, and M. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. In

Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, volume 37, 2007.

[30] J. B. D. Cabrera, L. Lewis, and R. K. Mehra. Detection and classification of intrusions and faults using sequences of system calls. *SIGMOD Records*, 30(4):25–34, 2001.

[31] S. Chakrabarti, S. Sarawagi, and B. Dom. Mining surprising patterns using temporal description length. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 606–617, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[32] P. K. Chan and M. V. Mahoney. Modeling multiple time series for anomaly detection. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 90–97, Washington, DC, USA, 2005. IEEE Computer Society.

[33] V. Chandola. http://www.cs.umn.edu/∼chandola/software.html, 2009.

[34] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection – a survey. Technical Report 07-017, University of Minnesota, Computer Science Department, August 2007.

[35] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection – a survey. *ACM Computing Surveys*, 41(3), July 2009.

[36] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection for discrete sequences: A survey. Submitted to TKDE, February 2009.

[37] V. Chandola, S. Boriah, and V. Kumar. A framework for exploring categorical data. In *Proceedings of the ninth SIAM International Conference on Data Mining*, 2009.

[38] V. Chandola, V. Mithal, and V. Kumar. A comparative evaluation of anomaly detection techniques for sequence data. In *Proceedings of International Conference on Data Mining*, 2008.

[39] V. Chandola, V. Mithal, and V. Kumar. Comparing anomaly detection techniques for sequence data. Technical Report 08-021, University of Minnesota, Computer Science Department, July 2008.

[40] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explorations*, 6(1):1–6, 2004.

[41] D. Cheboli, V. Chandola, and V. Kumar. Anomaly detection for time series data: A survey. Work in Progress, 2009.

[42] H. Cheng, P.-N. Tan, C. Potter, and S. Klooster. Detection and characterization of anomalies in multivariate time series. In *Proceedings of the ninth SIAM International Conference on Data Mining (SDM)*, 2009.

[43] C. Chow and D.-Y. Yeung. Parzen-window network intrusion detectors. In *Proceedings of the 16th International Conference on Pattern Recognition*, volume 4, page 40385, Washington, DC, USA, 2002. IEEE Computer Society.

[44] W. W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning*, pages 115–123, Tahoe City, CA, jul 1995. Morgan Kaufmann.

[45] D. Dasgupta and N. Majumdar. Anomaly detection in multidimensional data using negative selection algorithm. In *Proceedings of the IEEE Conference on Evolutionary Computation*, pages 1039–1044, Hawaii, may 2002.

[46] D. Dasgupta and F. Nino. A comparison of negative and positive selection algorithms in novel pattern detection. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, volume 1, pages 125–130, Nashville, TN, 2000.

[47] K. De Cock and B. De Moor. Subspace angles between arma models. *Systems and Controls Letters*, 46(4):265–270, July 2002.

[48] H. Debar, M. Dacier, M. Nassehi, and A. Wespi. Fixed vs. variable-length patterns for detecting suspicious process behavior. In *Proceedings of the 5th European*

Symposium on Research in Computer Security, pages 1–15, London, UK, 1998. Springer-Verlag.

[49] J. W. Demmel. *Applied Numerical Linear Algebra.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.

[50] D. Endler. Intrusion detection: Applying machine learning to solaris audit data. In *Proceedings of the 14th Annual Computer Security Applications Conference*, page 268. IEEE Computer Society, 1998.

[51] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection. In *Proceedings of Applications of Data Mining in Computer Security*, pages 78–100. Kluwer Academics, 2002.

[52] E. Eskin, W. Lee, and S. Stolfo. Modeling system call for intrusion detection using dynamic window sizes. In *Proceedings of DISCEX*, 2001.

[53] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[54] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1029, San Francisco, CA, 1993. Morgan Kaufmann.

[55] Z. Ferdousi and A. Maeda. Unsupervised outlier detection in time series data. In *Proceedings of the 22nd International Conference on Data Engineering Workshops*, page 121, Washington, DC, USA, 2006. IEEE Computer Society.

[56] G. Fernandez. *Data mining using SAS applications.* Chapman & Hall/CRC, Boca Raton, FL, USA, 2003.

[57] G. Florez-Larrahondo, S. M. Bridges, and R. Vaughn. Efficient modeling of discrete events for anomaly detection using hidden markov models. *Information Security*, 3650:506–514, 2005.

[58] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for unix processes. In *Proceedinges of the ISRSP96*, pages 120–128, 1996.

[59] S. Forrest, C. Warrender, and B. Pearlmutter. Detecting intrusions using system calls: Alternate data models. In *Proceedings of the 1999 IEEE ISRSP*, pages 133–145, Washington, DC, USA, 1999. IEEE Computer Society.

[60] A. J. Fox. Outliers in time series. *Journal of the Royal Statistical Society. Series B(Methodological)*, 34(3):350–363, 1972.

[61] M. Friendly. *Visualizing Categorical Data.* SAS Publishing, 2000.

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

[63] R. Fujimaki, T. Yairi, and K. Machida. An anomaly detection method for spacecraft using relevance vector learning. *Advances in Knowledge Discovery and Data Mining*, 3518:785–790, 2005.

[64] R. Fujimaki, T. Yairi, and K. Machida. An approach to spacecraft anomaly detection problem using kernel feature space. In *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 401–410, New York, NY, USA, 2005. ACM Press.

[65] P. Galeano, D. Pea, and R. S. Tsay. Outlier detection in multivariate time series via projection pursuit. Statistics and Econometrics Working Papers ws044211, Universidad Carlos III, Departamento de Estadïstica y Econometrïca, Sep 2004.

[66] P. Galeano, D. Pena, and R. S. Tsay. Outlier detection in multivariate time series via projection pursuit. Statistics and Econometrics Working Papers ws044211, Universidad Carlos III, Departamento de Estadïstica y Econometrïca, Sep 2004.

[67] V. Ganti, J. Gehrke, and R. Ramakrishnan. CACTUS–clustering categorical data using summaries. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 73–83, New York, NY, USA, 1999. ACM Press.

[68] B. Gao, H.-Y. Ma, and Y.-H. Yang. Hmms (hidden markov models) based on anomaly intrusion detection method. In *Proceedings of International Conference*

on Machine Learning and Cybernetics, pages 381–385. IEEE Computer Society, 2002.

[69] A. Ghosh, A. Schwartzbard, and M. Schatz. Learning program behavior profiles for intrusion detection. In *Proceedings of 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, pages 51–62, apr 1999.

[70] A. K. Ghosh, A. Schwartzbard, and M. Schatz. Using program behavior profiles for intrusion detection. In *Proceedings of SANS Third Conference and Workshop on Intrusion Detection and Response*, february 1999.

[71] D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: an approach based on dynamical systems. *The VLDB Journal*, 8(3):222–236, 2000.

[72] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000. Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215.

[73] G. H. Golub and H. Zha. The canonical correlations of matrix pairs and their numerical computation. Technical report, Stanford University, Stanford, CA, USA, 1992.

[74] F. A. Gonzalez and D. Dasgupta. Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4(4):383–403, 2003.

[75] S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[76] D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology.* Cambridge University Press, New York, NY, USA, 1997.

[77] R. Gwadera, M. Atallah, and W. Szpankowski. Reliable detection of episodes in event sequences. *Knowledge and Information Systems*, 7(4):415–437, 2005.

[78] R. Gwadera, M. Atallah, and W. Szpankowskii. Detection of significant sets of episodes in event sequences. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pages 3–10, 2004.

[79] D. Hawkins. Identification of outliers. *Monographs on Applied Probability and Statistics*, May 1980.

[80] S. Hawkins, H. He, G. J. Williams, and R. A. Baxter. Outlier detection using replicator neural networks. In *Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery*, pages 170–180. Springer-Verlag, 2002.

[81] Z. He, S. Deng, X. Xu, and J. Z. Huang. A fast greedy algorithm for outlier mining. In *Proceedings of 10th Pacific-Asia Conference on Knowledge and Data Discovery*, pages 567–576, 2006.

[82] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650, 2003.

[83] Z. He, X. Xu, and S. Deng. An optimization model for outlier detection in categorical data. In *Proceedings of International Conference on Intelligent Computing*, volume 3644. Springer, 2005.

[84] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[85] P. Hoffman and G. Grinstein. A survey of visualizations for high-dimensional data mining. In U. M. Fayyad, G. G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*, chapter 2, pages 47–82. Morgan Kaufmann, 2002.

[86] H. Hofmann. Exploring categorical data: interactive mosaic plots. *Metrika*, 51(1):11–26, 2000.

[87] S. A. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6(3):151–180, 1998.

[88] H. Hotelling. Relation between two sets of variates. *Biometrika*, 28:322–377, 1936.

[89] http://www.skaion.com/news/rel20031001.html. Skaion corporation. skaion intrusion detection system evaluation data.

[90] X. Hu, H. Qiu, and N. Iyer. Multivariate change detection for time series data in aircraft engine fault diagnostics. In *Proceedings of IEEE Internation Conference on Systems, Man and Cybernetics*, pages 2484–2489, October 2007.

[91] L. Huang, X. Nguyen, M. Garofalakis, M. Jordan, A. Joseph, and N. Taft. In-network pca and anomaly detection. Technical support, U.C. Berkeley, Berkeley, CA, 01/2007 2007.

[92] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.

[93] P. Huber. *Robust Statistics*. Wiley, New York, 1974.

[94] K. Ilgun, R. A. Kemmerer, and P. A. Porras. State transition analysis: A rule-based intrusion detection approach. *IEEE Transactions on Software Engineering*, 21(3):181–199, 1995.

[95] C. Jordan. Essai sur la géométrie à n dimensions. *Buletin de la Societe Mathematique de France*, 3:103–174, 1875.

[96] M. V. Joshi, R. C. Agarwal, and V. Kumar. Mining needle in a haystack: classifying rare classes via two-phase rule induction. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 91–102, New York, NY, USA, 2001. ACM Press.

[97] M. V. Joshi, R. C. Agarwal, and V. Kumar. Predicting rare classes: can boosting make any weak learner strong? In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 297–306, New York, NY, USA, 2002. ACM.

[98] M. Kano, S. Hasebea, I. Hashimotoa, and H. Ohno. A new multivariate statistical process monitoring method using principal component analysis. *Computers & Chemical Engineering*, 25(7–8):1103–1113, 2001.

[99] E. Keogh and T. Folias. The ucr time series data mining archive. http://www.cs.ucr.edu/eamonn/TSDMA/index.html, 2002.

[100] E. Keogh, J. Lin, and A. Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 226–233, Washington, DC, USA, 2005. IEEE Computer Society.

[101] 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*, 11(1):1–27, 2006.

[102] E. Keogh, S. Lonardi, and B. Y. chi' Chiu. Finding surprising patterns in a time series database in linear time and space. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 550–556, New York, NY, USA, 2002. ACM Press.

[103] E. Keogh, S. Lonardi, and C. A. Ratanamahatana. Towards parameter-free data mining. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215, New York, NY, USA, 2004. ACM Press.

[104] E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386, 2005.

[105] F. Knorn and D. Leith. Adaptive kalman filtering for anomaly detection in software appliances. *IEEE Conference on Computer Communications Workshops*, pages 1–6, April 2008.

[106] E. M. Knorr and R. T. Ng. Finding intensional knowledge of distance-based outliers. In *The VLDB Journal*, pages 211–222, 1999.

[107] E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000.

[108] Y. Kou, C.-T. Lu, and D. Chen. Spatial weighted outlier detection. In *Proceedings of SIAM Conference on Data Mining*, 2006.

[109] N. Kumar, V. N. Lolla, E. J. Keogh, S. Lonardi, and C. A. Ratanamahatana. Time-series bitmaps: a practical visualization tool for working with large time series databases. In *SDM*, 2005.

[110] V. Kumar. Parallel and distributed computing for cybersecurity. *Distributed Systems Online, IEEE*, 6(10), 2005.

[111] T. Lane and C. E. Brodley. Sequence matching and learning in anomaly detection for computer security. In Fawcett, Haimowitz, Provost, and Stolfo, editors, *Proceedings of AI Approaches to Fraud Detection and Risk Management*, pages 43–49. AAAI Press, 1997.

[112] T. Lane and C. E. Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information Systems and Security*, 2(3):295–331, 1999.

[113] J. Laurikkala, M. Juhola1, and E. Kentala. Informal identification of outliers in medical data. In *Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, pages 20–24, 2000.

[114] M. Lawrence, H. Wickham, D. Cook, H. Hofmann, and D. Swayne. Extending the GGobi pipeline from R. *Computational Statistics*, 2008.

[115] W. Lee and S. Stolfo. Data mining approaches for intrusion detection. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, TX, 1998.

[116] W. Lee, S. Stolfo, and P. Chan. Learning patterns from unix process execution traces for intrusion detection. In *Proceedings of the AAAI 97 workshop on AI methods in Fraud and risk management*, 1997.

[117] W. Lee and D. Xiang. Information-theoretic measures for anomaly detection. In *Proceedings of the IEEE Symposium on Security and Privacy*, page 130. IEEE Computer Society, 2001.

[118] C. S. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for svm protein classification. In *Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[119] J. Lin, E. Keogh, A. Fu, and H. V. Herle. Approximations to magic: Finding unusual medical time series. In *Proceedings of the 18th IEEE Symposium on Computer-Based Medical Systems*, pages 329–334, Washington, DC, USA, 2005. IEEE Computer Society.

[120] J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Min. Knowl. Discov.*, 15(2):107–144, 2007.

[121] R. P. Lippmann and et al. Evaluating intrusion detection systems - the 1998 darpa off-line intrusion detection evaluation. In *DARPA Information Survivability Conference and Exposition (DISCEX) 2000*, volume 2, pages 12–26. IEEE Computer Society Press, Los Alamitos, CA, 2000.

[122] G. Liu, T. K. McDaniel, S. Falkow, and S. Karlin. Sequence anomalies in the cag7 gene of the helicobacter pylori pathogenicity island. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 96 (12), pages 7011–7016, 1999.

[123] C.-T. Lu, D. Chen, and Y. Kou. Algorithms for spatial outlier detection. In *Proceedings of 3rd International Conference on Data Mining*, pages 597–600, 2003.

[124] J. Ma and S. Perkins. Online novelty detection on temporal sequences. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–618, New York, NY, USA, 2003. ACM Press.

[125] S. Ma and J. L. Hellerstein. Ordering categorical data to improve visualization. In *IEEE Information Visualization Symposium Late Breaking Hot Topics*, pages 15–18. IEEE, 1999.

[126] M. V. Mahoney and P. K. Chan. Learning nonstationary models of normal network traffic for detecting novel attacks. In *Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 376–385. ACM Press, 2002.

[127] M. V. Mahoney and P. K. Chan. Learning rules for anomaly detection of hostile network traffic. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, page 601. IEEE Computer Society, 2003.

[128] M. V. Mahoney and P. K. Chan. Trajectory boundary modeling of time series for anomaly detection. In *Proceedings of the KDD Workshop on Data Mining Methods for Anomaly Detection*, 2005.

[129] M. V. Mahoney, P. K. Chan, and M. H. Arshad. A machine learning approach to anomaly detection. Technical Report CS–2003–06, Department of Computer Science, Florida Institute of Technology Melbourne FL 32901, march 2003.

[130] D. Marchette. A statistical method for profiling network traffic. In *Proceedings of 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, pages 119–128, Santa Clara, CA, apr 1999.

[131] M. Markou and S. Singh. Novelty detection: a review-part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[132] M. Markou and S. Singh. Novelty detection: a review-part 2: neural network based approaches. *Signal Processing*, 83(12):2499–2521, 2003.

[133] C. C. Michael and A. Ghosh. Two state-based approaches to program-based anomaly detection. In *Proceedings of the 16th Annual Computer Security Applications Conference*, page 21. IEEE Computer Society, 2000.

[134] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Proceedings of the 7th International Conference on Artificial Neural Networks*, pages 999–1004, London, UK, 1997. Springer-Verlag.

[135] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM Press, 2003.

[136] M. E. Otey, A. Ghoting, and S. Parthasarathy. Fast distributed outlier detection in mixed-attribute data sets. *Data Mining and Knowledge Discovery*, 12(2-3):203–228, 2006.

[137] L. Parra, G. Deco, and S. Miesbach. Statistical independence and novelty detection with information preserving nonlinear maps. *Neural Computing*, 8(2):260–269, 1996.

[138] A. Patcha and J.-M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Comput. Networks*, 51(12):3448–3470, 2007.

[139] D. Pavlov. Sequence modeling with mixtures of conditional maximum entropy distributions. In *Proceedings of the Third IEEE International Conference on Data Mining*, page 251, Washington, DC, USA, 2003. IEEE Computer Society.

[140] D. Pavlov and D. Pennock. A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *Proceedings of Advances in Neural Information Processing*. MIT Press, 2002.

[141] K. Pearson. *On the Theory of Contingency and Its Relation to Association and Normal Correlation*. Dulau and Co., 1904.

[142] K. Pearson. On the general theory of multiple contingency with special reference to partial contingency. *Biometrika*, 11(3):145–158, 1916.

[143] C. Phua, D. Alahakoon, and V. Lee. Minority report in fraud detection: classification of skewed data. *SIGKDD Explorer Newsletter*, 6(1):50–59, 2004.

[144] B. Pincombe. Anomaly detection in time series of graphs using arma processes. *ASOR BULLETIN*, 24(4):2–10, 2005.

[145] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *Proceedings of ACM Workshop on Data Mining Applied to Security*, 2001.

[146] 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*, 369(2):677–696, 2006.

[147] Y. Qiao, X. W. Xin, Y. Bin, and S. Ge. Anomaly intrusion detection method based on *hmm*. *Electronics Letters*, 38(13):663–664, 2002.

[148] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[149] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438. ACM Press, 2000.

[150] U. Rebbapragada, P. Protopapas, C. E. Brodley, and C. Alcock. Finding anomalous periodic time series. Submitted to the Machine Learning Journal, 2008.

[151] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: learning probabilistic automata with variable memory length. *Machine Learning*, 25(2-3):117–149, 1996.

[152] G. Rosario, E. Rundensteiner, D. Brown, M. Ward, and S. Huang. Mapping nominal values to numbers for effective visualization. *Information Visualization*, 3(2):80–95, 2004.

[153] V. Roth. Outlier detection with one-class kernel fisher discriminants. In *NIPS*, 2004.

[154] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection.* John Wiley & Sons, Inc., New York, NY, USA, 1987.

[155] S. Salvador and P. Chan. Learning states and rules for time-series anomaly detection. Technical Report CS–2003–05, Department of Computer Science, Florida Institute of Technology Melbourne FL 32901, march 2003.

[156] S. Salvador and P. Chan. Learning states and rules for detecting anomalies in time series. *Applied Intelligence*, 23(3):241–255, 2005.

[157] R. Saunders and J. Gero. The importance of being emergent. In *Proceedings of Artificial Intelligence in Design*, 2000.

[158] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, 2001.

[159] S. Shekhar, C.-T. Lu, and P. Zhang. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In *Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 371–376, New York, NY, USA, 2001. ACM Press.

[160] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of 3rd IEEE International Conference on Data Mining*, pages 353–365, 2003.

[161] D. Snyder. Online intrusion detection using sequences of system calls. Master's thesis, Department of Computer Science, Florida State University, 2001.

[162] X. Song, M. Wu, C. Jermaine, and S. Ranka. Conditional anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 19(5):631–645, 2007.

[163] C. Spence, L. Parra, and P. Sajda. Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, page 3, Washington, DC, USA, 2001. IEEE Computer Society.

[164] A. Srivastava. Nasa dashlink. https://dashlink.arc.nasa.gov, 2008.

[165] A. N. Srivastava. Discovering system health anomalies using data mining techniques. In *Proceedings of 2005 Joint Army Navy NASA Airforce Conference on Propulsion*, 2005.

[166] I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.

[167] G. W. Steward and J. guang Sun. *Matrix Perturbation Theory.* Academic Press, 1990.

[168] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Proceedings of the 5th IEEE International Conference on Data Mining*, pages 418–425, Washington, DC, USA, 2005. IEEE Computer Society.

[169] P. Sun, S. Chawla, and B. Arunasalam. Mining for outliers in sequential databases. In *In SIAM International Conference on Data Mining*, 2006.

[170] P. Sutherland, A. Rossini, T. Lumley, N. Lewin-Koh, J. Dickerson, Z. Cox, and D. Cook. Orca: A visualization toolkit for high-dimensional data. *Journal of Computational and Graphical Statistics*, 9(3):509–529, 2000.

[171] D. F. Swayne, D. T. Lang, A. Buja, and D. Cook. GGobi: evolving from XGobi into an extensible framework for interactive data visualization. *Computational Statistics & Data Analysis*, 43(4):423–444, 2003.

[172] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[173] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, Boston, MA, 2006.

[174] 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*, pages 535–548, 2002.

[175] D. Tax and R. Duin. Data domain description using support vectors. In M. Verleysen, editor, *Proceedings of the European Symposium on Artificial Neural Networks*, pages 251–256, Brussels, April 1999.

[176] D. Tax and R. Duin. Support vector data description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.

[177] D. M. J. Tax. *One-class classification; Concept-learning in the absence of counter-examples*. PhD thesis, Delft University of Technology, June 2001.

[178] H. Teng, K. Chen, and S. Lu. Adaptive real-time anomaly detection using inductively generated sequential patterns. In *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy*, pages 278–284. IEEE Computer Society Press, 1990.

[179] J. Theiler and D. M. Cai. Resampling approach for anomaly detection in multispectral images. In 230-240, editor, *Proceedings of SPIE 5093*, 2003.

[180] R. S. Tsay, D. Pea, and A. E. Pankratz. Outliers in multivariate time series. *Biometrika*, 87(4):789–804, 2000.

[181] R. S. Tsay, D. Peja, and A. E. Pankratz. Outliers in multivariate time series. *Biometrika*, 87(4):789–804, 2000.

[182] R. Vilalta and S. Ma. Predicting rare events in temporal domains. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, page 474, Washington, DC, USA, 2002. IEEE Computer Society.

[183] L. Wei, E. Keogh, and X. Xi. Saxually explicit images: Finding unusual shapes. In *Proceedings of the Sixth International Conference on Data Mining*, pages 711–720, Washington, DC, USA, 2006. IEEE Computer Society.

[184] L. Wei, N. Kumar, V. Lolla, E. J. Keogh, S. Lonardi, and C. Ratanamahatana. Assumption-free anomaly detection in time series. In *Proceedings of the 17th international conference on Scientific and statistical database management*, pages 237–240, Berkeley, CA, US, 2005. Lawrence Berkeley Laboratory.

[185] A. S. Weigend, M. Mangeas, and A. N. Srivastava. Nonlinear gated experts for time-series - discovering regimes and avoiding overfitting. *International Journal of Neural Systems*, 6(4):373–399, 1995.

[186] G. M. Weiss and H. Hirsh. Learning to predict rare events in event sequences. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *Proceedings of 4th International Conference on Knowledge Discovery and Data Mining*, pages 359–363, New York, NY, 1998. AAAI Press, Menlo Park, CA.

[187] H. Wickham, M. Lawrence, D. Cook, A. Buja, H. Hofmann, and D. Swayne. The plumbing of interactive graphics. *Computational Statistics*, 2008.

[188] Q. Wu and Z. Shao. Network anomaly detection using time series analysis. In *Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services*, page 42, Washington, DC, USA, 2005. IEEE Computer Society.

[189] J. Yang and W. Wang. Cluseq: Efficient and effective sequence clustering. In *Proceddings of International Conference on Data Engineering*, pages 101–112, 2003.

[190] Y. Yang, X. Guan, and J. You. CLOPE: a fast and effective clustering algorithm for transactional data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 682–687, New York, NY, USA, 2002. ACM.

[191] D. Yankov, E. J. Keogh, and U. Rebbapragada. Disk aware discord discovery: Finding unusual time series in terabyte sized datasets. In *Proceedings of International Conference on Data Mining*, pages 381–390, 2007.

[192] N. Ye. A markov chain model of temporal behavior for anomaly detection. In *Proceedings of the 5th Annual IEEE Information Assurance Workshop*. IEEE, 2004.

[193] X. yun Chen and Y. yan Zhan. Multi-scale anomaly detection algorithm based on infrequent pattern of time series. *Journal of Computational and Applied Mathematics*, 214(1):227–237, April 2008.

[194] M. J. Zaki and M. Peters. CLICKS: Mining subspace clusters in categorical data via k-partite maximal cliques. In *ICDE 2005: Proceedings of the 21st International Conference on Data Engineering*, pages 355–356, 2005.

[195] H. Zare Moayedi and M. Masnadi-Shirazi. Arima model for network traffic prediction and anomaly detection. *International Symposium on Information Technology*, 4:1–6, Aug. 2008.

[196] A. J. Zeevi, R. Meir, and R. Adler. Time series prediction using mixtures of experts. In *Advances in Neural Information Processing*, volume 9. MIT Press, 1997.

[197] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, 2004.

[198] X. Zhang, P. Fan, and Z. Zhu. A new anomaly detection method based on hierarchical *hmm*. In *Proceedings of the 4th International Conference on Parallel and Distributed Computing, Applications and Technologies*, pages 249–252, 2003.