Anomaly Detection on the Digital Video Broadcasting System

Xiaoran Liu

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

Anomaly Detection on the Digital Video Broadcasting System

by

Xiaoran Liu

in partial fulfillment of the requirements for the degree of

Master of Science
in Computer Science

at the Delft University of Technology,
to be defended publicly on Monday December 19, 2016 at 11:30 AM.

Student number:	4417410	
Supervisor:	Dr. ir. S. E. Verwer,	TU Delft
	Dr. ir. D. S. (Dmitri) Jarnikov,	Irdeto B.V.
Thesis committee:	Prof. dr. J. V. D. Berg,	TU Delft
	Dr. S. E. Verwer,	TU Delft
	Dr. D. Tax,	TU Delft
	Dr. D. S.(Dmitri) Jarnikov,	Irdeto B.V.
	MSc. Q. Lin,	TU Delft

TU Delft Delft University of Technology

PREFACE

Firstly, I wish to express my gratitude to both my supervisor, Dr.Sicco Verwer and Dr.D.S.(Dmitri) Jarnikov for offering me the opportunity of working on this interesting project. Their continued support and kind help motivated me thorough the whole project.

I would like to appreciate the time and effort that Rob Schipper, Qin Lin spent with me on this project. Their sufficient and knowledgeable background gives me strong support about technical understanding. This thesis would not have been successful without their timely help and useful suggestions. I would like to thank my friends Gergely, Zelin Wang and Wei Xie who give me strong mental support when I feel confused and depressed.

I would also like to thank all my colleagues in Irdeto who are very friendly and kind. They give me a lot technical support and advice for me to improve. This first working experience helps me developing my further career interest.

Finally, I would like to thank my parents and all my family back in China for their support and encouragement throughout my course of study at Delft University of Technology. With shifts of major, I grow a lot during the past two years in Delft University of Technology. Except knowledge, I have more practical experiences and my character also benefits from the living in the Netherlands.

Thanks for the people who accompany with me during the past colourful time.

Xiaoran Liu
Delft, November 2016

Contents

LIST OF TABLES

List of Figures

[1] https://en.wikipedia.org/wiki/Deterministic_finite_automaton

1

INTRODUCTION

Nowadays proliferation of accessible data stream happens through various networks such as broadcasting, web page, mobile phone, etc. There is a lot of useful information hidden in the fast and huge data to be further learned and analyzed. The big size and continuously real-time updating character of data streams pose challenges for studying and mining. The techniques with strong computing power and adaptive capability are highly required for handling multiple tasks.

As one of the data mining goals, anomaly detection is a common goal shared by different domains. Anomalies are generally defined as an error or as an unexpected event in reality. Based on the application domains, an anomaly can be the cancer diagnostic of a patient, a fraud of credit card, and a fatal problem of a running system. Though anomalies are defined differently per domain, the commonly recognized statistical assumption is that in data space, an anomaly tends to emerge with a low probability. In addition to its distributional property, features like duration, repetition, and severity are used as supplemental evidence. Anomalies can also be contextual, which corresponds and to an unexpected event or abnormal behavior given a realistic situation. This research focuses on implementing data-driven techniques to detecting anomalies in the monitoring transport stream data generated by Digital Video Broadcasting system (DVB). Anomaly detection is of great importance to the DVB system from both the provider and consumer perspective. From the providers' perspective, anomaly detection not only guarantees the condition of the system, but also gives instinct opinions during the data analysis process, such as the quantity of consumers, the efficiency of bandwidth, and so on. From the users' perspective, detecting anomalies in the transport stream protect the program quality and users' experience. In this research, the anomaly detection task starts from the providers' scope.

DVB standards have been adopted in Europe as open standards for the digital broadcasting system for a long period. For a variety of subsystems, e.g. Satellite, Cable, Terrestrial television, and Microwave. DVB standards define both physical and data link layer for these systems. Furthermore, DVB standards define how the multiple program data is distributed and transported among the protocols in the format of MPEG Transport Stream (MPEG-TS). Figure 1.1 illustrates the block diagram of the MPEG-2 and DVB layered structure [7].

MPEG-TS defines the standard container format for transmission and storage of data stream through the DVB system [1]. Packetized Elementary Stream (PES) contains video, audio, data, and teletex, which is part of the MPEG-TS. Except for PES, both Service Information (SI) and Program Service Information (PSI) are defined into table types and they are segmented into sections before being inserted into MPEG-TS according to the DVB-MPEG standard. [2]. In this research, MPEG-TS is the source of our input. Instead of starting from user experience scope, it reveals the condition of the operating DVB system.

1.1. RESEARCH SUMMARY

This research targets the problem of detecting and even predicting anomalies in the DVB system. The inputs are varied data sets collected from the TS, which is clearly explained in Chapter 3. The input data is not the MPEG-TS but its monitored recording. In other words, the collected data is the monitoring of MPEG-TS, which contains important features such as scrambling transactions, bit rate of media data in one packet, and

[1] http://www.afterdawn.com/glossary/term.cfm/mpeg2_transport_stream
[2] https://www.dvb.org/resources/public/standards/a38_dvb-si_specification.pdf

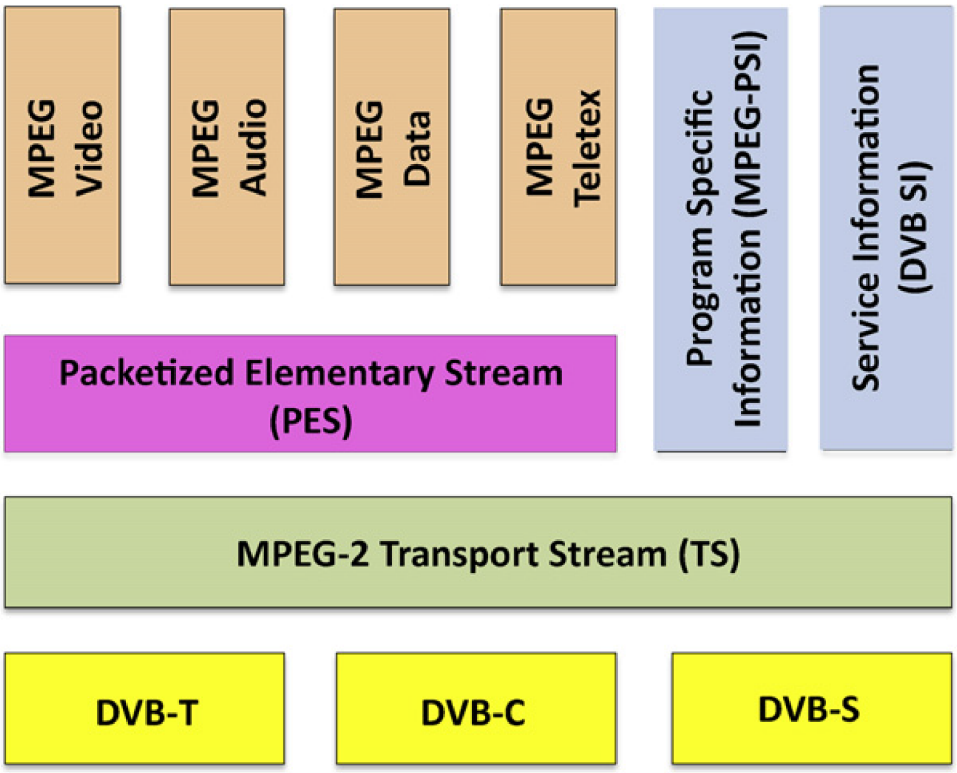


Figure 1.1: High level overview of the MPEG-2 and DVB layered structure. MPEG Transport stream is distributed and transported among the protocols in the system structure. [1]

etc. It reflects the operation condition of the DVB system. Figure 1.2 is the example of input data. It clearly records the features and important data of the TS.

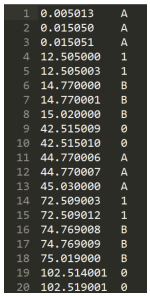
Figure 1.2: An example of the original collected data. For time instance, the records of events and features occur are toggled.

In this research, the state-of-art data mining techniques e.g. grammar learning, principle component analysis, and similarity-based techniques have been tested over different data sets. There are three kinds of data generated from the collected data, which are discrete event sequences, univariate time series, and nontraditional multivariate time series. The discrete event sequences and univariate time series are extracted from the collected data. The nontraditional multivariate time series is the collected data itself. Figure 1.3 gives examples of these three kinds of data sets. The examples show the initial data sets before any further preprocessing. All these data sets have taken multiple and organized preprocessing steps in order to adapt the analysis requirement. The expected result is identifying anomalies correctly and giving warnings.

1.1.1. CHALLENGES

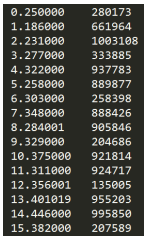
The research challenges root from both the nature of the collected data and the anomaly detection task itself.

- It is hard to define anomalies without expert knowledge in a complex system. From a statistical perspective, an anomaly appears with low probability. However, from the contextual view of understanding data, the definition of the anomaly may be different from the assumption of the rarity. And the definition of the anomaly may have changes when the system is under different conditions. Background knowledge of DVB also influences the classification between anomaly types and the setting of accuracy measurement.
- The test data set has imbalanced data because anomaly occurrences are relatively rare. There is the statistical assumption that an anomaly in the data space always indicates the lowest probability in a data



```
1  0.005013   A
2  0.015050   A
3  0.015051   A
4  12.505000  1
5  12.505003  1
6  14.770000  B
7  14.770001  B
8  15.020000  B
9  42.515009  0
10 42.515010  0
11 44.770006  A
12 44.770007  A
13 45.030000  A
14 72.509003  1
15 72.509012  1
16 74.769008  B
17 74.769009  B
18 75.019000  B
19 102.514001 0
20 102.519001 0
```

(a) example of discrete event sequence



```
0.250000   280173
1.186000   661964
2.231000   1003108
3.277000   333885
4.322000   937783
5.258000   889877
6.303000   258398
7.348000   888426
8.284001   905846
9.329000   204686
10.375000  921814
11.311000  924717
12.356001  135005
13.401019  955203
14.446000  995850
15.382000  207589
```

(b) example of univariate time series

```
"TIM":"2016-04-21 16:17:47.706","IND":  1,"PID": 3583,"PIH":"0dff","SCR":"O-U"}
"TIM":"2016-04-21 16:17:47.706","IND":  2,"PID": 3583,"PIH":"0dff","SCR":"U-O"}
"TIM":"2016-04-21 16:17:47.713","IND":  0,"PID":  701,"PIH":"02bd","CASys.ID":"0d95","Table-ID":"84","Reserved":"7","Length":"05
"TIM":"2016-04-21 16:17:47.713","IND":  1,"PID":  701,"PIH":"02bd","CASys.ID":"0d95","Table-ID":"84","Reserved":"7","Length":"05
"TIM":"2016-04-21 16:17:47.713","IND":  2,"PID":  701,"PIH":"02bd","CASys.ID":"0d95","Table-ID":"84","Reserved":"7","Length":"05
```

(c) example of collected data/nontraditional multivariate time series

Figure 1.3: example of three kinds of input data

distribution. However, for a classifier-based technique, the class imbalance will influence the degree of accuracy. For a high accuracy of finding anomalies, false positives (identifying normal behavior as abnormal behavior) are acceptable to a certain extent. In contrast, treating anomalies as normal behavior sometimes can be extremely costly, for example, in medical diagnosis.

- The monitoring data has a large amount of dimensions, which poses a burden for representation, transformation, and further detection.
- The data set contains mixed variable types, i.e., numeric and categorical. Furthermore, they have diverse ranges. It is of great importance to assemble features in a uniform fashion, which is more efficient to adapt different techniques.
- The precision level of the monitoring data is microseconds, but the monitoring data is logged when the desired information pops up in the TS. In other words, the attributes of the monitoring data are highly asynchronous.

1.1.2. RESEARCH OBJECTIVES

We aim to answer the following research questions:

- The data tested in this research is not a traditional time series. It contains varied feature types and the sampling time among features is not equal. How to overcome obstacles rooted from its characters?
  Apparently, a preprocessing step with multiple methods and appropriate order is required. The steps have to be processed in the way which does no harm to the data. In order to deal with this, an initial observation of the raw data should focus on reasonable transformation components. Small steps during the transformation should be considered without causing more bias or raising conflicts in the original data space.
- Without prior knowledge of the system, how to deal with such huge amounts and high-dimensional data sets? How to overcome the challenge raised by asynchronized attributes?
  When facing up with such complex system and without efficient expert knowledge, an initial study of the data set should be completed, such as discovering the relationship and difference among attributes.

Data-driven dimension reduction techniques and expert knowledge are both helpful to shed light on feature engineering.

- With prior knowledge of the system, how to combine this information to build a model with more explanatory power for starters? How to make advantage of this information adequately for classifying anomaly types?
  With the help of expert knowledge for the system, a much lower dimension of data can be extracted. Machine learning techniques that result in white-box models will help to analyze the underlying mechanisms of the system.

1.1.3. RESEARCH CONTRIBUTIONS

This thesis is an attempt to find an appropriate solution to analyze and detect anomalies in Digital Video Broadcasting systems from monitoring data of the transport stream. The contributions are as follows,

- In this research, a real-time time automaton used has been appropriately implemented for detecting anomalies in discrete event sequences, which can also be further implemented for predicting. It has strong interpretable capabilities for event transitions with time constraints. The advantage is that it considers both the syntactic and timed information for the discrete event sequences.
- To address the challenges of un-synchronization and mixed feature types in the data set, the data has been reasonably transformed to high dimensional multivariate time series. Not only overcomes the challenges from the data characters, but also provides the possibility for testing varied state-of-art detection techniques. It is beneficial for starters who want to know this system, which can also be referenced for other researchers when facing up with high dimension data from an unfamiliar system. Besides, this research provides feasible detection techniques and make a comparison with consideration of varied targeted test data type.
- For Digital Video Broadcasting system, we provide suggestions and solutions to deploy white-box and black-box models for timed strings and multivariate time series data respectively. It could be furthermore developed and manipulated with varied initial settings of the system. This is also beneficial for experts in this field to dig more useful information with strong background knowledge.

1.2. OUTLINE

The thesis is organized as follows. Chapter 2 is the background and related work. Chapter 3 deploys the data collection process and the characters of varied data sets. Chapter 4 explains different detection algorithms applied in this thesis. Chapter 5 explains the experiment process and discusses the experimental results. Chapter 6 proposes conclusion remarks and points out the future work.

2

Background and Related Work

2.1. Anomaly Definition

Anomaly detection refers to finding unexpected or inconsistent patterns in the data, which typically indicates that unusual events happen. Anomalies can also be considered as outliers, surprises, exceptions, noises and novelties according to different application domains[2]. Anomaly detection is a typical data mining task that aims at detecting an event or judging the condition, such as cyber intrusion, credit card fraud, monitoring system health, medical diagnostics and etc.

Anomalies can be separated into three groups [2], which consider both its character and practical meaning.

- **Point Anomalies**
  If an individual data instance can be considered as anomalous with respect to the rest of the data, then it is termed as a point anomaly. This uses statistical detection techniques. This kind of anomaly is the simplest type and could be detected by visualizing the entire data space.

- **Contextual Anomalies**
  Contextual anomalies refer to a conditional anomaly, [8], defining from the context environment of a dataset. The data consists of contextual and behavioural attributes. Contextual anomalies have been most commonly found in time series data. Take the classical example of one-year monthly temperature in location *A*. The contextual attribute is the month, which decides the data position in the sequence, and the behavioural attribute is the temperature value. Supposed the lowest temperature occurs twice, the one happened in the months of winter is normal, but the one happens in summer months would be anomalous. And also, for another location *B*, the definition of contextual anomalies will probably be different. Figure 2.1 is an example of contextual anomaly. Same temperature occur at both time t_1 and t_2, however, t_2 is an anomaly while t_1 does not.

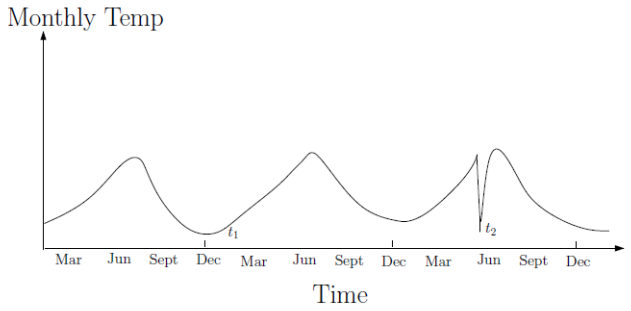


Figure 2.1: Contextual anomaly t_2 in a temperature time-series.
Time t_1 has the same temperature, but in different context and hence is not considered as anomaly. [2]

- **Collective Anomalies**
  Each individual data point itself is normal when compared to the entire dataset, but the collection of them or consistency of appearance is an anomalous situation. The data instances, in this case, should be related with each other. Collective anomalies are studied in several fields, like sequence data[9], graph data[10], and spatial data [11].

Though point anomalies and collective anomalies occur in different dataset environments. There exists a transition between these and contextual anomaly. With respect to a contextual environment, point anomalies and collective anomalies can also become a contextual anomalies. The anomaly type also influences the choice of detection technique. Figure 2.2 is an example of collective anomaly, which shows a human electrocardiogram output. The red region denotes an anomaly because the same low value exists for an abnormally long time. The low value itself is not an anomaly.

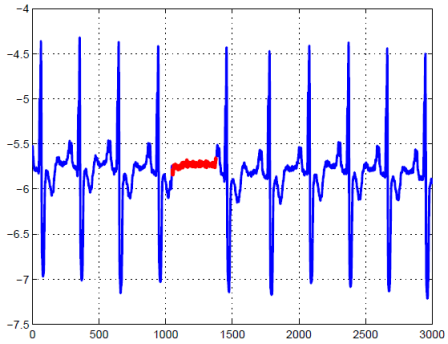


Figure 2.2: Collective anomaly corresponding to an Atrial Premature Contraction in an human electrocardiogram output. [3]

2.2. ANOMALY DETECTION TECHNIQUES

To detect anomalies, the first challenge is to define the anomaly and the second is to apply reasonable techniques. With varied anomaly definitions and types among application domains, corresponding techniques for anomaly detection are also varied. Different detection techniques hold different assumptions with anomaly construction and intrinsically applied restrictions. For statistical techniques, the low probability of distribution could be the anomaly definition, but for classification-based technique, the distribution density may not be persuasive as anomaly definition. For supervised learning techniques, the availability of labeled training data is also an influential factor.

Based on the survey of anomaly detection [2], anomaly detection techniques can be categorized into the following groups.

- **Statistical techniques**
  The assumption of statistical techniques is to indicate anomalies as outliers in a data distribution. In other words, the anomalies(minority) should appear in a low probability while the normality(majority) occurs with a high probability from the particular model. There is an essential prerequisite of these techniques that a statistical model or function can fit a conceptual reference paradigm, such as Gaussian distribution, Poisson distribution, and linear regression. The majority of these techniques are targeting at single dimension or univariate data[12]. Statistical techniques can be further classified into two groups considering whether it is parametric. For non-parametric techniques, the model structure is obtained from given data, such as based on its probability density. One classical example is the Grubbs' test for detecting anomalies in the univariate dataset, which is based on the assumption of Gaussian distribution. The Grubbs' test can typically standard for the statistics-based anomaly detection method with the requirement of normal distribution [13]. Figure 2.3 is an example from Tietjen and Moore, which applied Grubbs' test for detecting the outlier. The normal assumption in this case is reasonable with the exception of the maximum value, while the maximum value turns out to be an outlier though Grubbs' test.

  Statistical techniques have assumptions for detecting anomalies in a dataset. The first limitation is that a prior knowledge or reasonable guess about the data distribution is required, which is not always

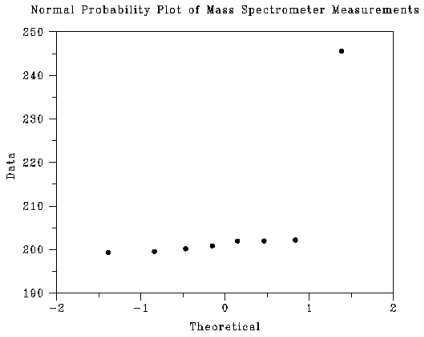


Figure 2.3: The plots of 8 mass spectrometer measurements on a uranium isotope.
The maximum value is an outlier in this case. [4]

available in practice. Secondly, a huge amount of data is required for parameter inference. Besides, anomalous data characters sometimes cannot be completely explained by statistical features, such as dynamic behaviours with timely features cannot be captured by the model.

- **Classification-Based techniques**
  Classification is to identify to which category a new instance belongs, based on a known classifier or rule obtained from training data instance. Based on the quantity of classes, different classification techniques can be implemented, and some of them can cover both one-class and multi-class situation with setting adjustment. Common supervised learning techniques are involved, such as Neural Networks, Support Vector Machines. In [14], one-class Support Vector Machines is applied for finding targeted outliers in the hand written digits. Figure 2.4 is a visualized example of detecting anomalies though classification. The quantity of the normal classes is decided by the labels. Once an instance is not classified as normal by any of the classifiers, it should be considered anomalous. The training

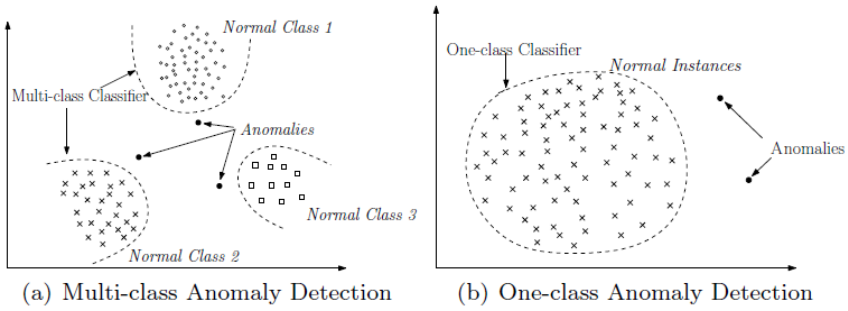


Figure 2.4: Example of applying classification for anomaly detection.
The left one is multi-class, and the other one is one-class. [2]

data instances should be adequate and accurate for obtaining good classifying capability. When detecting anomalies, classification-based techniques can even distinguish anomaly types when facing multi-class cases, but they highly depend on the availability of accurate labels. Also, the inadequate labels of anomalies will probably influence the test accuracy.

- **Nearest Neighbour-Based techniques**
  The nearest neighbour-based techniques roots from the density in data space, anomalies are far from their closest neighbours while normal instances occur in dense neighbours. The distance, which is the similarity between two data instances can be computed in different methods. Euclidean distance is one of the most common distance computing methods, which is also implemented in this research. The amount of nearest neighbour and the parameters of pair distances for one data instance may give different hints about density distribution. Some researches defines the anomalous score of an instance based on the distance to its k^{th} nearest neighbours. The setting of k and the manipulation of distance

is varied among applications. For example, Zhang and Wang compute the anomaly score of a data instance as the sum of its distance from its k nearest neighbours [15]. Some researches define the anomalies based on the density of nearest neighbours. The data lies in a low density neighbourhood is more anomalous than the one lying in a dense density neighbourhood. But if the data set has regions of varying densities, density-based techniques will perform poorly. Figure 2.5 is a good example of this case. Both p_1 and p_2 are anomalies, but p_1 may be detected and p_2 will not be distinguished from C_2. For every instance in C_1, its nearest neighbour distance is apparently larger than the nearest neighbour distance between p_2 and instances in C_2. Some algorithms such as Local Outlier Factor proposed by Breunig et al.[16] are capable of capture both anomalies with the help of additional parameters. The

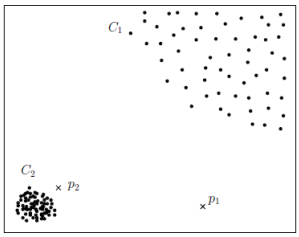


Figure 2.5: Local density-based techniques have advantages over global-density based techniques. [2]

significant advantage of nearest-neighbour-based techniques is that these are naturally unsupervised and totally data driven, which have no assumption or requirement of the data structure. Except for the distance measure, these techniques need adequate close neighbours for normal cases. If the anomalies form a dense space in the dataset, these techniques may fail to identify anomalies.

- **Clustering-based techniques**
  Clustering is to group objects so that the ones within the same group/cluster will be more similar with each other than with the ones from other clusters [17]. The basic concept of clustering satisfies the requirement of similarity measurement issue. The assumption of an anomaly in such techniques is easy to be proposed. Either the anomalies do not belong to any known clusters or the anomaly is far away from its closest cluster centroid but normal instances lie close to their closest cluster centroids respectively. In case the anomalies form clusters by themselves, the third assumption is that anomalies either belong to small or sparse clusters while normal instances are not. In [18], the normal cases are distinguished from anomalies in the training data. Then use frequent item-set mining and clustering-based technique detecting anomalies. In this research, K-means clustering has been implemented for testing the anomalous instances.

  Clustering-techniques solves the similarity measurement issue itself, but its efficiency is highly dependent on capturing the structure of normal instances. And also every instance is forced to be assigned to some cluster, which may enlarge the cluster and anomalies may be treated as normal cases.

- **Spectral techniques**
  Spectral techniques look for an approximation of the data using a combination of attributes which capture the main variability in the original data. In other words, data can be embedded into a lower dimensional subspace where normal instances and anomalies are significantly different. Spectral techniques can handle high dimension datasets because it reduces dimensionality automatically. Principal Component Analysis(PCA) is quite a common tool for projecting data into a lower dimensional space[19]. In this research, PCA has also been used for testing anomalies in the multivariate time series. [20] applied PCA to detect anomalies in astronomy catalogs. Dutta et al. used uniform sampling for developing a distributed PCA algorithm, whose performance was compared with a random projection based approach. Their algorithm could effectively explore more outliers, which was the common expectation in astronomy. However, the communication cost needed to be reduced.

  In general, the advantage of spectral techniques is that it handles the high dimension problem and can be viewed as an unsupervised setting, which has no high requirement for prior knowledge about the data. However, it is only useful when normal and anomalous instances can be easily separated in the lower dimensional subspace of the original data.

There are other rules to separate anomaly detection techniques, for instance, the principle of machine learning categories can be applied, which is based on the availability of training data labels. And the categories of anomaly detection techniques are also different when focusing on narrowly applied field. No matter how the categories differentiate, the core is to understand the unusual behavior, and then find appropriate techniques with accuracy and consistency.

2.3. CURRENT RESEARCH

The majority research related to DVB system monitoring is mainly focusing on the users' experience and efficiency improvement. Predicting the packet loss based on MPEG-2 is a hot research area too. And the research scope is so wide because different related application domains keep varied datasets and goals. The difference of researches starts from the data collection and preprocessing process. In [1], Francesco and his colleagues especially focus on DVB card-sharing traffic monitoring over the internet. DVB card-sharing is an illegal activity, which means that multiple people can access to the Pay-TV though only one of them has the authority. This is a critical issue related to the security problem. They would like to detect such unlawful behaviors without causing harm to the payload at the same time. Their research addresses the potentials of wavelet analysis, especially for characterizing the observations. They observe the MPEG-2 signal traffic flow on multiple scales of time and frequency, similarly with the data collected in this research, and abstract wavelet-related properties. Based on wavelet-related features, they implement a Support Vector Machines (SVMs) as a binary classification for detecting anomalies. They provide SVMs in order to deal with dynamic complex model. Their ideas demonstrate the possibility of a pure operational use of machine learning techniques together with concepts derived from the dynamics of complex systems. They combine these two in order to develop dynamic deterministic models given qualitative and quantitative observations. Their approach avoids relying on the packet contents and does no harm to the payload. With the help of wavelet analysis and then binary classification, they provide support for defeating unlawful activities online. In [21], Aninda, Alexender, and Amir apply Recurrent Neural Networks (RNN) to predict MPEG-coded Video Source Traffic, targeting at improving dynamic bandwidth allocation and multimedia quality-of-service (QoS) control strategies over the network. In the network area, how to efficiently and fairly utilize the bandwidth is a hot and critical research topic. Their predication of multimedia traffic is based on the frame definitions from MPEG encoded standards. Their training process use the visual object plane (VOP) as the target, which is the objects in MPEG-4 image. The training is on MPEG-4 traces, while the performance test is applied on Yoo's MPEG-1 traces. They use a feedforward multilayer perceptron and a recurrent multilayer perceptron neural networks respectively to construct two kinds of neuro predictors. One neuro predictor is for use in single-step (SS), and the other one is for multistep (MS) prediction horizons. Based on their test results, these two developed predictors are generic to some extent, and they can be used to predict the video source traffic for a wide range of MPEG-coded video streams without any tuning. From [22], Martin et al. present a new two-level Markov model for TS packet loss over UDP/IP and this model can also model the performance of forward error correction on improved application design. They propose this model to model the packet delivery process accurately so that further analytic and simulation studies can be brought out. The model uses both packet loss and delay information to better understand the state of the network. The delay information is for classifying the packet loss reason. Their model involves the consideration of the network congestion situation and performs very well compared with past models. From [23] A SVMs classifier is applied for classifying the visibility of TS packet losses, specially in the SDTV and HDTV MPEG-2 compressed video streams. They separated the frames into reference frames and non-reference frames. And the kinds of frame loss are entire and partial frame loss. The test indicates not only the classification result, but also provides some insights. For example, TS packet losses are more visible for HD than for SD according to the test result.

Currently, there is small amount research related to detecting anomalies of the DVB system, particularly from the provider scope. This research is starting from the provider scope instead of users' respective. This research has varied anomalies related to the dataset collected, rather than a specific problem to be detected in the system. In this research, spectral-based, clustering-based, and nearest neighbour based techniques have been tested for the multivariate time series. Except for the categories in the earlier section, for univariate time series and discrete event sequence, grammatical inference, as a specified ruled based technique, is also implemented to detect anomalies.

3

DATA PREPROCESSING

3.1. DATA COLLECTION

MPEG-TS consists of a sequence of packets, which are the basic units containing data and information. Note that not all packets in MPEG-TS contain informatical contents. Some null packets are inserted with random bytes by a multiplexer to keep packet size 188 bytes constantly. For most useful packets, their structure is shown as Figure 3.1 [1].

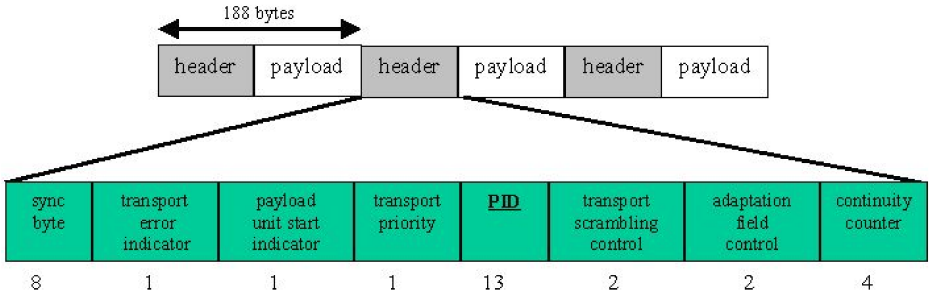


Figure 3.1: The structure of none empty MPEG-TS packet, which consists of head and body components. Video, audio and meta-data are segmented into frames, which are inserted into this packet.

Packets with fixed length of 188 bytes include 4 bytes header and 184 bytes payload. The components in the packet header are important with a different indication. The Packet Identifier (PID) of 13 bytes is the identifier of this TS packet. The function of PID is to distinguish packets and linked with PSI/SI tables. And it is the key to mapping MPEG-TS packets with program tables. More details of DVB and MPEG-TS standards are not included in the aims of this paper. Readers who interested in more formal definitions are referred to the standardization reports available on the DVB project web site. [2]

In this research, the data is collected by a Data Collector, which is an application for the logging all Conditional Access System related data of any Transport Stream. And the collected data is presented in JSON format. Every 10 minutes a new file is being used, and every day the collection of files is being sanitized. This initial data mainly contains two kinds of logging along time: one is the static recording of events, such as PID, Table-ID, which has been used for identification and link to certain components inside the TS packet. The other one is the dynamic monitoring attributes or characters of the happening event, such as the appearance of the program information table and the bit rate of the transport stream. These monitoring observations are also the references as measurements for DVB system condition. Each measurement keeps its individual characters with a different number of attributes. Not all the values inside the time series monitoring stream are continuously over time. Both the observations of happening events and their corresponding attributes have been used for anomaly identification and further analysis. The precision scale is the microsecond. Table 3.1 gives a brief description of properties of these observations.

[1] http://mhpkdbwiki.s3.unidue.de/mhpkdbwiki/index.php/Transport_Stream
[2] https://www.dvb.org/standards

Name	Type	Properties
Timestamp	Index	Mark time
PID(Packet Identifier)	Static	Identification of packet
CAsys ID (Conditional Access System ID)	Static	Identification of conditional system
Table ID	Static	Identification of program information table/sections
CAT(Conditional Access Table)	Event	Table ID:1, appears every 2 seconds, contains information on the CA Systems present in the current transponder
PAT (Program Association Table)	Event	Table ID:0, PID:0, Length:21, contains information on the services available in the current transponders
PMT (Program Map Table)	Event	Table ID:2, contains information on the service
SDT(Service Description Table)	Event	Table ID:42, appears every 2 seconds, provide additional Service Information (name and number) in the PMT fields
TDT (Time Delivery Table)	Event	Table ID:70, appears every 30 seconds
TOT (Time Offset Table)	Event	Table ID:73, appears every 30 seconds
ECM(Entitlement Control Message)	Event	Encrypted control word
EMM(Entitlement Management Message)	Event	Decryption authority of control word
SCR(Scrambling Transition)	Event	Scrambling process
Length	Statistics	Section features, limited to 12 bits in hexadecimal
Address Value	Statistics	Section features, EMM address values
Address Length	Statistics	For different CA systems different lengths can be found
Bitrate	Statistics	PEG-TS packet/section features

Table 3.1: description of data components

The values of these monitoring measurements contribute to an untraditional multivariate time series. These measurements have different types, some are categorical and some are numeric. Besides, these measurements are toggled when occur, which means their sampling times are not the same. With the known background knowledge, three types of data set applied in this research are given in the later sections.

3.2. DISCRETE EVENT SEQUENCE

The discrete event sequence in this research is abstracted from the initial data set, with the combination of two related measurements. It is based on the targeted meta-data from single TS packets. The categorical attributes/events related to the targeted meta-data form a discrete event sequence, there is a certain behavioral loop inside the sequence. Figure 3.2 is part of discrete event sequence. The cycling loop is fixed with four symbols. The precision level of time is the microsecond, but the toggling does not keep stable time interval though the time interval for typical events' transition is certain. The four alphabets are "A","B","0","1", which are two pairs. The event "0" and "1" represent encryption mode, following the event "A" and "B" respectively. The frequency between ECM can be 10 seconds, 20 seconds and 30 seconds. The frequency between event "0" and "1" is around 30 seconds in this data set. The event "A" and "B" stand for encryption information/order, meaning ECMs in the system. In real life, the access to digital video data stream is provided only to those with valid decryption smart-cards. A combination of scrambling and encryption is helpful to achieve this goal. The data stream is scrambled with a 48-bit secret key, called the control word. Encryption is used to protect the control word during transmission to the receiver: the control word is encrypted as an entitlement control message (ECM). The control word must be informed slightly in advance so that no viewing interruption occurs. This is also reflected in the time difference in the data set. Just after receiving ECMs (event "A" and "B"), corresponding encryption mode happens (event "0" and "1"). The amount of paired events is fixed in one loop, if the amount of event "A" is n, then so is event "B". But the total amount of one event loop is not fixed. Thus, the amount of events in the loop is not appropriate to detect whether the next event or the next

loop is missing, regardless of the time interval.

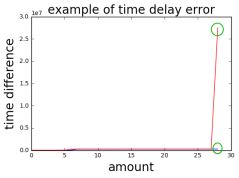
```
1  0.005013   A
2  0.015050   A
3  0.015051   A
4  12.505000  1
5  12.505003  1
6  14.770000  B
7  14.770001  B
8  15.020000  B
9  42.515009  0
10 42.515010  0
11 44.770006  A
12 44.770007  A
13 45.030000  A
14 72.509003  1
15 72.509012  1
16 74.769008  B
17 74.769009  B
18 75.019000  B
19 102.514001 0
20 102.519001 0
```

Figure 3.2: A Brief Example of Discrete Event Sequence

For this data set, there are two kinds of anomalies. The first one is shared, which is missing data. The other one is timing error, such as time delay. Figure 3.3 gives example of the two anomalies. Figure 3.3a shows where event "A" is missing. It should exist between event "0" and event "1". Figure 3.3b explains the time delay situation. The red line stands for the abnormal trend and the blue line stands for the normal situation respectively. The y-axis is the value of time difference of one event transition, which is increased to 10^6 larger for visualization. It is shown in the figure that the anomaly has a significant delay of 30 seconds. Based on the

```
331 1634.785012 B
332 1634.785013 B
333 1635.035004 B
334 1635.035005 B
335 1662.620016 0
336 1662.625021 0
337 1692.625031 1
338 1692.630011 1
339 1694.785008 B
340 1694.785009 B
341 1695.035004 B
```

(a) Example of event lost anomaly. Based on the certain events' loop, it is obvious that event "A" is missing, which should exist between event "0" and event "1".



(b) Example of Time Delay happened in the discrete event sequence. The y-axis stands for time difference of one certain event transition. The blue line is normal while the the red line is abnormal situation. As the blue line demonstrated, this certain event transition mostly happens around 0 second. It is obvious that the abnormal case has a serious delay. The big difference of these two values are circled in green.

Figure 3.3: example of two anomalies

discrete event sequences, two kinds of grammatical inference techniques have been applied, one is based on N-grams, the other one is RTI+ algorithm[24], which provides learnt models with strong explanatory power.

3.3. TIME SERIES DATA

A time series $T = (t_1, t_2, \ldots t_n)$ is a an ordered sequence of time units t_i, with i from 1 to n, with same sampling interval between consecutive observations. A set of variables or measurement $v = (o_1, o_2, \ldots o_n)$ are the corresponding collection of observations or underlying process, which can be numeric or categorical. Each measurement o_i has attributes $u_1, u_2, \ldots u_k$, which can be represented in vectors. Time series data can be separated into univariate time series and multivariate time series based on the amount of dimensions. If the amount of variable is single, i.e. $k = 1$, then the time series is considered as univariate time series. A multivariate time series data stream is a time sequence of data elements with multiple attributes, i.e. $k > 1$ [25].

In this thesis, we discuss both univariate and multivariate time series as per distinct scenarios. The multivariate time series is also the original data set obtained from the data collector. It does not fit the time series definition perfectly, which adds challenge to anomaly detection task.

3.3.1. UNIVARIATE TIME SERIES

The univariate time series in this research is extracted from one numeric attribute of unique TS packet, which is its packet bandwidth. The toggling time is around per second, and the error is within the microsecond(10^{-6}) range. The toggling time can be treated as sampling time directly. This univariate time series fits the traditional univariate time series definition. It keeps equal sampling time and has numeric value changing along with time sample. The given univariate time series is already labeled as normal and abnormal. From theoretical scope, point, contextual, and collective anomalies could all appear, while in our research, there are two kinds of abnormal cases. One anomaly is the numeric value crossing the limitation range, the other one is missing data. Figure 3.4 is the plot of this univariate time series which contains anomaly of information missing. For around 430 minutes there is no toggling of value, which is a serious error.

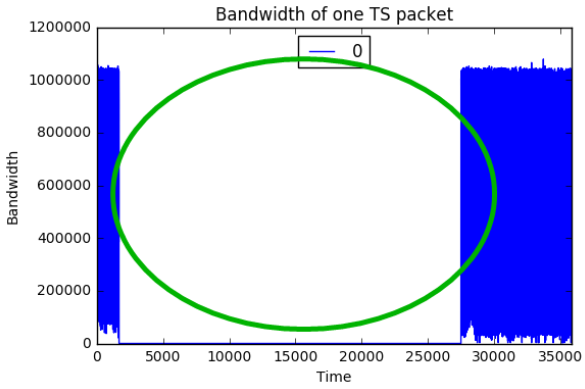


Figure 3.4: Example of Data Lost in Univariate Time Series. The y-axis stands for the bit rate of one TS packet, and the x-axis is time scale. where a long period of empty value reflects the severity of this problem. The anomaly is marked by the green circle.

3.3.2. MULTIVARIATE TIME SERIES

In general, multivariate time series data has the following characters: the first one is the high dimension. Time series stream has other features varied among domains and these characters are mainly generated from the patterns of time series sequences. From the financial domain, the seasonal character is of great importance for patterns mining and forecasting. From the environmental domain, climate records like wind speed are noisy due to turbulence nature [26][27]. In [28], the periodical and synchronous features have been addressed and associated with each other as the classification principle for multivariate time series. The periodical character indicates the repetition of patterns in time sequences and the synchronous character indicates the aligned pattern among variables in time series. Not all the multiple variables are synchronized and they may have different initial time point. The correlated relationship among variables inside one multivariate time series stream is also a typical character. The unique value of multiple variables is independent with each other such as the data generated from a sensor, which means they may not have the similar pattern or trend, though they may have dependence among each other.

The multivariate time series data in this work is collected from continuously monitored containing all concerned attributes in Table Table 3.1. It has the following specified characteristics

- High dimension and large size of data.
- Data has both numeric and categorical features.
- Numeric features keep value range with huge difference.
- Data is not synchronized, i.e. each dimension of data has its own sampling interval.

The first character is relatively common, but the other characters make the initial data set not a classical multivariate time series. The characters of original multivariate time series add the analysis and computing burden. Figure 3.5 is an interval example of the initial data set, with numeric value stands for meaningful categorical features. It is obvious that the values of varied attributes lying sparsely and a large amount of discontinuous value exist. A preprocessing step which could transform the data for further analysis is required.

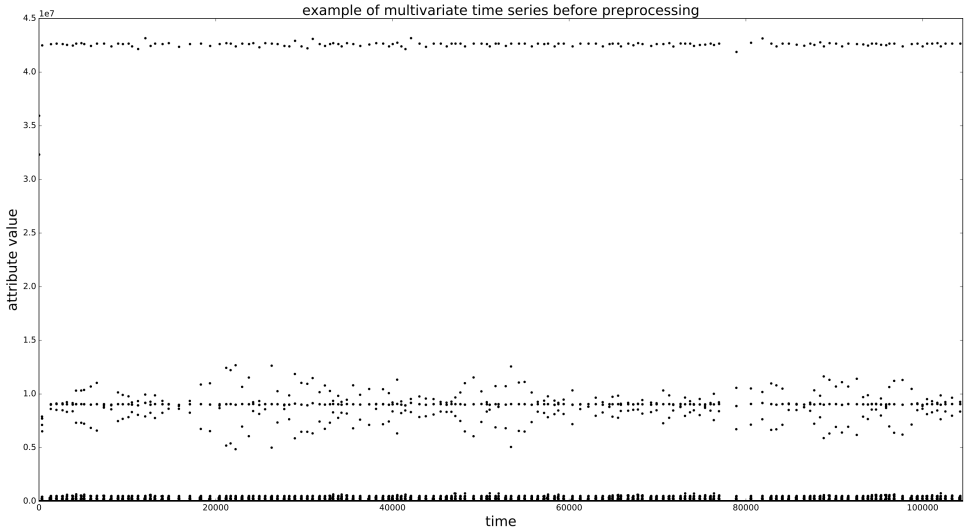


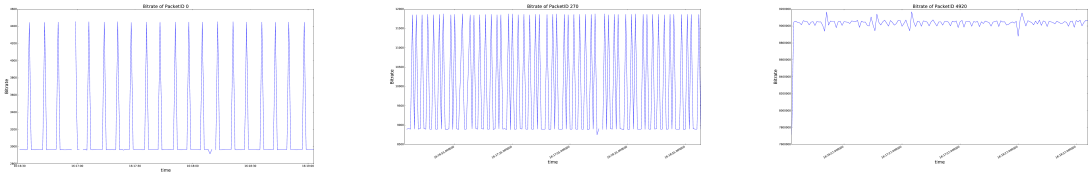
Figure 3.5: an interval example of multivariate time series before preprocessing, each vertical line stands for the data obtained at that time instance, the dots stands for the value of numeric attributes and categorical meaning at each time instance

Without reducing the dimensions, we firstly transform the rest characters to make the original data set as a multivariate time series.

The preprocessing step is completed as following:

- The initial step for transforming the data, the minimum sampling interval is picked up to overcome the unequal sampling frequency problem. Minimum sampling interval makes sure that all the observations have been captured, though empty values appear for some features with larger sampling interval before. The precision level is microsecond in the initial data set. The minimum sampling interval decided is per second, make all the numeric features continuous over time.

- For the empty cells inside categorical features, meaningful numeric value standing for empty has been inserted to make these features become continuous. A linearly normalized process has been implemented for numeric dimensions because the numeric values lie in ranges with huge gap. It scales values to lie between zero and one inside one dimension so that all the numeric data distributes without a huge gap in magnitudes. Figure 3.6 is an example of the numeric attributes, whose values are fluctuating in the different value range. Because the TS packet contains different variables to be monitored, their bitrate varied a lot, such as Packet 4920 should contain lots image and sound data.



(a) PID 0 (b) PID 270 (c) PID 4920

Figure 3.6: The varied value ranges and patterns of numeric features among TS packets. There is huge gap among different numeric features, which should be normalized.

- Besides, the categorical features have been transformed through implementing one-hot encoding, which transforms each categorical with m possible values to m binary features. In digital circuits, one-hot refers to a group of bits among which the legal combinations of values are only with a single high (1) bit and all the others low (0)[29]. One-hot encoding is often used for indicating the state of a machine. In Natural Language Processing, a one-hot vector is a $1 \times N$ matrix (vector) which has been implemented

for distinguishing each word in a vocabulary from every other word in the vocabulary. The vector consists of 0s in all cells but exists a single 1 in a cell used uniquely to identify the word. In data science and machine learning, it is used to encode categorical integer features using a one-hot aka one-of-K scheme with the assumption that the features are discrete (categorical). For example, a data set of ['house','car','people'] can be transferred to a matrix as

$$\begin{matrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{matrix}$$

through one-hot encoding. Or suppose the feature is 'color', containing five values which are 'black','whi te','blue','red','yellow'. One-hot encoding transforms these values into binary features, which can be represented by a 5×5 matrix and each vector in it has unique 1 activated. The length of each vector is the amount of the classes in one feature. And also for further analysis, the matrix (each vector) obtained by one-hot encoding should not be influenced by normalization of numeric attributes. The disadvantage of implementing one-hot encoding in our research is that the quantity of dimensions at each time unit has been increased, but it provides the appropriate way to adjust categorical features prepared for detecting anomalies.

Respectively, Figure 3.7 is an interval example of original data set after transformation. The sampling time in Figure 3.7 is every second, in total three minutes. The plot is the result after the preprocessing step of transformation and linearly normalization. Each point within a reasonable horizontal range stands for the values of the monitoring measurement attribute or the meaningful values of categorical measurement attributes, contributing to the dimension of the data space.

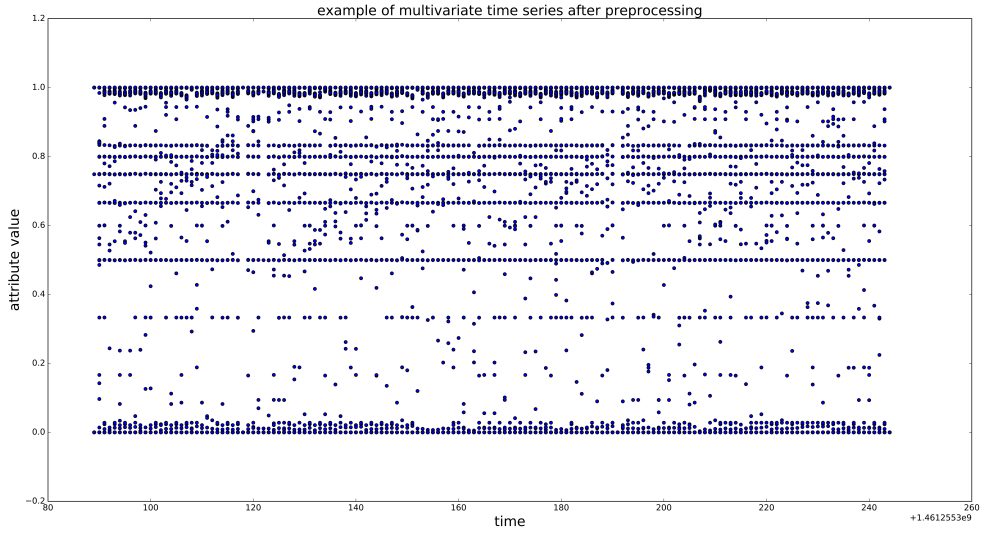


Figure 3.7: an interval example of multivariate time series after preprocessing step, x-axis stands for continuous time, y-axis stands for per time value. There are multiple dots, showing the value of each attribute/feature at this time instance.

4

TECHNIQUES

Chapter 4 discusses the anomaly detection techniques involved in this research, which are N-gram, RTI+, Grammarviz, Principal Component Analysis, k-means clustering, and k-Nearest Neighbour. According to different data sets, varied techniques have been applied during the experiment process. In this chapter, the background knowledge and building details of these techniques and models are explained. N-gram and RTI+ rooted from grammar inference are applied for discrete event sequence. Consisting of Symbolic Aggregate Approximation and context-free grammar, Grammarviz is applied for detecting anomalies in univariate time series. Principal Component Analysis, k-means clustering and k-Nearest Neighbour are applied for detecting anomalies in multivariate time series.

4.1. GRAMMAR INFERENCE

Grammar inference, also know as grammar induction, is the process of automatically inferring a grammar by examining the sentences of an unknown language, which used successfully in a variety of fields such as pattern recognition, natural language processing, machine learning, etc. The grammar is alternatively as an automaton or finite state machine obtained from a set of observations. To be described concisely, "grammatical inference is the learning task to identify a correct grammar for the (unknown) target language, given a finite number of examples of the languages" [30]. The grammar refers to a finite representation of possibly an infinite set of observations, such as strings, trees or graphs. The language refers to the possibly infinite set of instances. Most inference techniques begin with the given set of observations and makes a series generalizations from them, either completed by state-merging (in finite automata), or nonterminal merging (in context-free grammars) [31]. A grammar does not describe the meaning of the strings or what can be done with them in whatever context, but the form of strings. A grammar is defined as a set of production rules for strings, which describe how to form strings from language's alphabet that are valid according to the language's syntax[1].

The Chomsky hierarchy is a containment hierarchy of classes which classifies grammar levels [32]. It is described by Noam Chomsky in 1956 and Figure 4.1 is the set inclusions described by it.

Four levels are described in the Chomsky hierarchy:

- Type-0 grammars (unrestricted grammars)

  As its name, unrestricted grammars has no restrictions on either left or right side of the grammar production. It generate the languages which can be recognized by a Turing machine. And it is most general in this level structure.

- Type-1 grammars (context-sensitive grammars)

  Context-sensitive grammars have rules of the form $\alpha A\beta \rightarrow \alpha\gamma\beta$. In this form, A belongs to a set of nonterminal symbols and α,β,γ strings of terminals and/or nonterminals. The string γ must be nonempty. Context-sensitive grammars produce languages that can be recognized by a linear bounded automaton.

[1] The syntax of a computer language is the set of rules that defines the combinations of symbols that are considered to be a correctly structured document or fragment in that language.

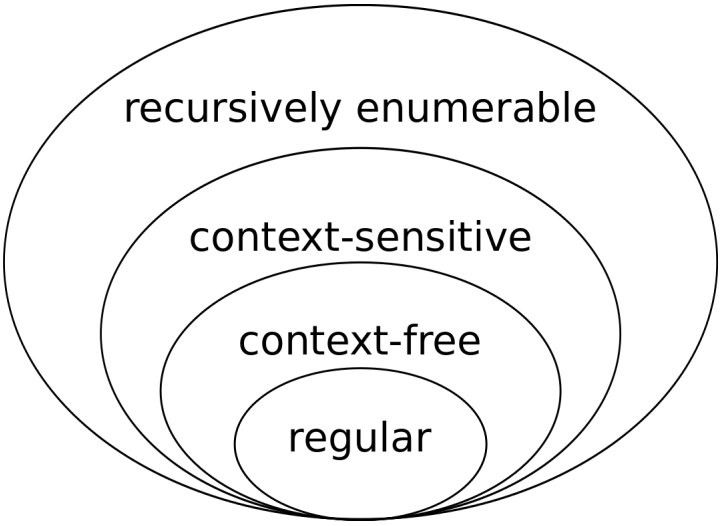


Figure 4.1: Chomsky hierarchy structure, which classifies grammar levels

- Type-2 grammars (context-free grammars)

  Just as its name, the rules produced by context-free grammars can be applied regardless of context. Its production rule is simple replacement. A nonterminal symbol is always on the left-side of production rule. A subset of deterministic context-free language are the theoretical basis for the phrase structure of most programming languages.

- Type-3 grammars (regular grammars)

  Regular grammars can be separated to left and right regular grammar. For both these two grammars, a single nonterminal symbol is on the left-side and the right-side consists of a single terminal symbol. For right regular grammar, there can be a single nonterminal symbol following the single terminal symbol. These two grammars generate the same languages, however, if left-regular rules and right-regular rules are combined, the language need no longer be regular. The regular grammars produce languages that can be decided by a finite state automaton.

Based on the Chomsky hierarchy, it is not hard to understand that automata are used as models for describing grammar rules and language behaviors. Except the structure classification of grammars, the learning models, complexity and difficulty of grammar inference itself also raise attention.

In this research, based on two kinds of data sets, n-grams, the real-time automaton (a variant of DFA), and context-free grammar have been applied for detecting anomalies. The following subsections discuss n-gram, real-time automata and context-free grammar respectively.

4.1.1. N-GRAM

An n-gram is a contiguous sequence of n items from a sequence of text or speech whose model has been applied for predicting next item in continuous sequence. The items are called "gram". An n-gram model is similar with a $(n-1)$ order Markov model, which is one kind of probabilistic language models. It has been widely used for many fields such as DNA sequencing analysis in bio-informatics field. The sequential consideration of n-gram model just fits the core idea of categorical attributes in time series, which covers both the appearance of events but also the order. In another word, a simple n-gram model gives the probability of next item based on some number of previous items, the probability distribution is obtained in the training set. An n-gram distribution is computed by sliding a fixed size window though data stream and counting the amount of occurrences of each "gram"[33].

In this research, the n-gram model has been applied to get the probability of transition among events in the discrete event sequence. Table 4.1 is an example of how n-gram applied over DNA sequences. The discrete event sequence has unequal sampling time in this research. In this case,if N-gram is applied directly over the discrete event sequence, the time constraints will be ignored. We would like the N-gram could cover time related issues. Therefore, the events in our data set should be adjusted to have equal sampling time. The sampling time is set as every second, and the empty instances appeared are filled as earlier event. In real life,

Sample Sequence	Unigram	Bigram	Trigram
...AGCTTCGA...	...,A,G,C,T,T,C,G,A,...	...,AG,GC,CT,TT,TC,CG,GA,...	...,AGC,GCT,CTT,TTC,TCG,CGA,...

Table 4.1: example of DNA sequencing[2]

this step is also explainable. Each event could be viewed as an order. Once received the order, the system will change its state and remain this state until receiving the next or new order. Though n-gram is less sensitivity in context modification, however, it could find the inner relation among events and orderly patterns in our case. And it is very simple and direct to tell whether there is an anomaly or not.

There is one disadvantage of n-gram is that it lacks long range of dependency. The next output is based on the $(n-1)$ tokens. Combined with the cycling event loop phenomenon in discrete event sequence, if an entire loop is lost, it would be hard for n-gram to detect. The setting of n is also an influential factor. Two training data sets within same time period have been compared, one is the normal data set, and the other one contains timing error as an anomaly. The result is the probability distribution of the following item/event transition, which has been compared between two data sets.

4.1.2. RTI+

Based on the classification of grammars, regular grammars can be represented by Deterministic Finite Automata, which are also common models for discrete event. However, the data obtained from real world is not always only in the type of discrete event sequence. Such as our test data, which contain time values of event transitions, and DFA has limited capability of handling this obstacle. As its variant, Timed Automata (TA) is more helpful than DFA in this case. In this research, an algorithm called RTI+ has been implemented to model the cycling behavior of discrete events with time constraints. RTI+ stands for Real-Time Identification from Positive Data, which is capable of identifying sufficiently large Deterministic Real-Time Automata (DRTAs) in order to identify the real-time systems [24].

Before the time constraints involved, the Deterministic Finite Automaton (DFA) is introduced here, as it is a common model for handling discrete event sequence. And the main components of a red-blue framework, aiming at state merging in DFAs [34], is also implemented in the later process. The DFA is formally defined as following:

Definition 1 (DFA) A deterministic finite state automaton (DFA) M is a 5-tuple $\langle Q, \Sigma, \delta, q_0, F\rangle$, where Q is a finite set of states, Σ is a finite set of input symbols called the alphabet, δ is a transition function ($\delta : Q \times \Sigma \rightarrow Q$) representing one state/symbol to the next, q_0 is an initial or state belonging to Q, and F stands for a set of accept states included in Q.

A DFA example is demonstrated in Figure 4.2 in the format of state diagram. States S_0, S_1, and S_2 are denoted by circles in the graph. When reading a symbol, a DFA moves deterministically from one state to another, followed the arrow. A finite sequence of 0s and 1s are the input of this automaton. In practical world, events

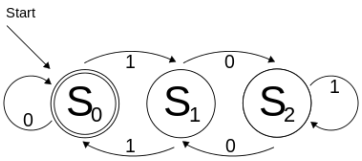


Figure 4.2: An example of a DFA. The state S_0 is the start state and a finite sequence of 0s and 1s is accepted.[3]

always happen with timing, so does the discrete event sequence in this research. But time constraints are implicit in DFAs. A Deterministic Real-Time Automaton (DRTA), as a variant of DFA, has the capability of handling time constraints, has been raised up. Except for the normal state transitions used in DFAs, time transitions are managed too. A DRTA is also a 5-tuple similar with DFA, the only difference is that its transition function δ is also a tuple. One event transition $\delta \in \Delta$ in DRTA is a tuple $\langle q, q', a, [n, n']\rangle$, where $q, q' \in Q$ are the source and target states, $a \in \Sigma$ is a symbol, and $[n, n']$ is a time guard. And for the same symbol, the same source state and overlapping delay guards in one DRTA, there is only one transition, this is why it is deterministic. Figure 4.3 [5] demonstrates an example of DRTA. A DRTA accepts and rejects timed strings based on both the event symbols and time values. From Figure 4.3, it accepts (a, 4)(b, 3) (state sequence: left → bottom → top) and (a, 6)(a, 5)(a, 6) (left → top → left → top), but rejects (a, 6)(b, 2) (left → top → reject) and

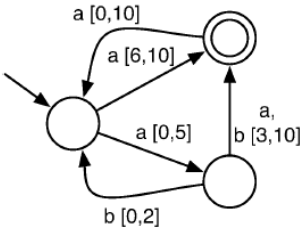


Figure 4.3: An example of a DRTA. The leftmost state is the start state, indicated by the sourceless arrow. The topmost state is an end state, indicated by the double circle. Every state transition contains both a label and a delay guard.[5]

(a, 5)(a, 5)(a, 6) (left → bottom → top → left). The RTI algorithm represents real-time identification, is used for identifying DRTAs [5]. It is based on the evidence-driven state-merging (EDSM) algorithm in a red-blue framework, which is one efficient and good performing algorithms for identifying DFAs[35]. A state-merging algorithm starts from constructing a tree automata from the input, and then merges the states of this tree. For each sample in this tree, there is a path from the root node to its node. The node in which a positive or negative example ends is marked positive (accepting) or negative (rejecting), respectively. The merge of two states combines the states into one when all input transitions of both states point to this new node and this new node covers the output transitions of both nodes. Such a merge processing is only allowed if the states are consistent, i.e. when no positive node is merged with a negative nodes [24]. Except of simply merging states, RTI is capable of handling the time values and delay guards through splitting process. First assume that every delay guard contains all delay values, and then split them based on timed evidence. A *split s(d,t)* of transition d with clock guard $g = [t_1, t_2]$ at time t divides d into two new transitions d' and d", with delay guards $g' = [t1, t]$ and g" $= [t + 1, t2]$ respectively.

The difference between RTI and RTI+ is that the setting of identifying DRTAs in RTI+ is based on positive data rather than labeled data. In order to identify a DRTA from positive data S_+, a probability distribution for timed strings with a DRTA structure shall be modeled. Adapting RTI to identify such *probabilistic DRTAs* (PDRTAs), how to present the probability of observing a certain timed event (a, t) given the current state q of the PRDTA needs to be decided. For every state q in the PRDTA, two distributions are essential, one for the possible symbols, and one for the possible time values $Pr(T = t \mid q)$. These two distributions determine the probability of the next state $Pr(X = q' \mid q)$. The distribution over events $Pr(S = a \mid q)$ is based on the standard generalization of the Bernoulli distribution. For example, given current state q, every symbol a has some probability $Pr(S = a \mid q)$, and the equation exists: $\Sigma_{a\in\sigma} Pr(S = a \mid q) = 1$. The distribution over time $Pr(T = t \mid q)$ is completed through model of histograms. Given a fixed number of bins H, every bin $[v, v'] \in H$ is an interval in N. Inside the bins, distributions are modeled uniformly, for example, for all $[v, v'] \in H$ and all $t, t' \in [v, v']$,$Pr(T = t \mid q) = Pr(T = t' \mid q)$. Naturally, it holds that all these probabilities sum to one:$\Sigma_{t\mathbb{N}} Pr(T = t \mid q) = 1$. Besides, the assumption here is that the bins are specified beforehand, for example by a domain expert, or by performing data analysis. It is effective to use histograms for modeling the time distribution[24].To make these two distributions independently is a common practice[36], and it avoids increasing the size of the model by a polynomial factor. The definition of PDRTA is as following:

Definition 2(PDRTA) A probabilistic DRTA (PDRTA) A is a quadruple $\langle A', H, S, T\rangle$, where $A' = \langle Q, \Sigma, \Delta, q_0\rangle$ is a DRTA without final states, H is a finite set of bins (time intervals)$[v, v']$, $v, v' \in \mathbb{N}$, known as the histogram, S is a finite set of symbol probability distributions $S_q = Pr(S = a \mid q) \mid a \in \Sigma, q \in Q$,and $T$is a finite set of time-bin probability distributions $T_q = Pr(T \in h \mid q) \mid h \in H, q \in Q$.

The probability of an observation (a, t) given that the current state q is defined as

$$Pr(O = (a, t) \mid q) = Pr(S = a \mid q) \times Pr(T = t \mid q)$$

and the probability of next state q given that the current state q is defined as

$$Pr(X = (q' \mid q) = \sum_{\langle q,q',a,[v,v']\rangle\in\Delta} \sum_{t\in[v,v']} Pr(O = (a, t) \mid q)$$

Figure 4.4 shows an example a PDRTA A, which can be used as predictor of timed events. Let $H = \{[0, 2]; [3, 4]; [5, 6]; [7, 10]\}$ be the histogram. In every bin the distribution over time values is uniform. For example, the

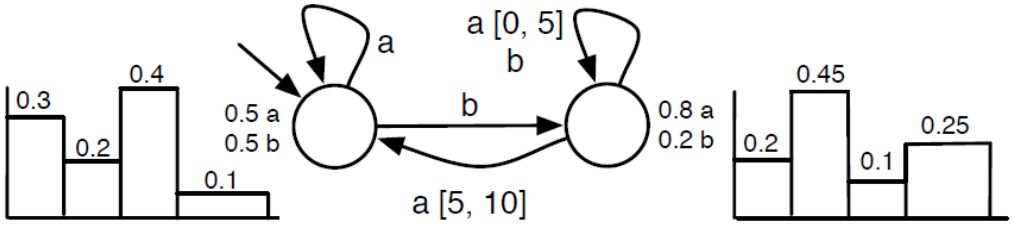


Figure 4.4: A probabilistic DRTA. Every state is associated with a probability distribution over events and over time. The distribution over time is modeled using histograms. The bin sizes of the histograms are predetermined but left out for clarity.[5]

probability of $(a,3)(b,1)(a,9)(b,5)$ is $Pr((a,3)(b,1)(a,9)(b,5)) = 0.5 \times \frac{0.2}{2} \times 0.5 \times \frac{0.3}{3} \times 0.8 \times \frac{0.25}{4} \times 0.5 \times \frac{0.4}{2} = 1.25 \times 10^{-5}$.

RTI+ has the same framework and operations as RTI, except the timed evidence. Apparently, in RTI, the timed evidence is based on the amount of positive and negative instances which end in the same state. However, only positive examples are used in RTI+ regardless which states these examples ends in. RTI+ initials an augmented prefix tree acceptor(APTA) without accepting and rejecting states, because it is generated from positive data. And then follows the same two steps as RTI, which is merge and split. And then a likelihood-radio test has been used both as an evidence value and as a consistency check. The likelihood-ratio test [37] is a common tool for testing nested hypothesis. A hypothesis H is called nested within another Hypothesis H' if the possible distributions under H form a strict subset of the possible distributions under H'. In another word, H can be created by constraining H'. Given two hypothesis H and H', H is nested in H', and a data set $\{S_+\}$, the likelihood-ratio test statistic is computed by

$$LR = \frac{likelihood(\{S_+, H\})}{likelihood(\{S_+, H'\})}$$

where likelihood is a function which returns the *maximized likelihood* of a data set under a hypothesis. For example,$likelihood(S_+, H)$ is the maximum probability (with optimized parameter settings) of observing S_+ under the assumption that H was used to generate the data. Let H and H' have n and n' parameters respectively. Because H is nested H', $likelihood(S_+, H')$ is always greater than $likelihood(S_+, H)$. Hence, the likelihood-ratio LR is a value between 0 and 1. When the difference between n and n' is bigger, the likelihood under H' can be optimized more and LR will be closer to 0. The likelihood-ratio test can test whether this increase in likelihood is statistically significant or not. The test compares the value $-2\ln(LR)$ to a χ^2 distribution, with degrees of freedom equal to the difference in parameters of H' and H $(n' - n)$[38]. The result of this comparison is a p-value, when it is high, it indicates that H is a better model. It is straightforward to use the likelihood-ratio test for both state-merging and transition-splitting. Take the state-merge for example. Take two PDRTAs, one A is the result after merging the states and the other A' is before merging. Clearly, A is nested in A'. Thus the maximized likelihood of S_+ under A and A' needs to be computed for test. Since PDRTAs are deterministic, the maximized likelihood can be computed simply by setting all the probabilities in the PDRTAs to their normalized counts of occurrence in S_+. Similar does the likelihood-ratio test for splitting transitions. The obstacle needs to overcome is that the likelihood-ratio test does not perform well when there are not adequate used parameters. When there are many unused parameters, the increase in the number of parameters will usually not be significant. So null-hypotheses, such as merging states, is easier to be accepted. This flaw is artificially fixed by *pooling* the bins of the histogram and symbol distributions if the frequency of these bins in both states is less than 10. Pooling is to combine the frequency of two bins into a single bin.

For RTI+, the null-hypothesis means that two states are the same. Whether to accept or reject the hypothesis depends on p-value, if p-value is less than 0.0500, then reject the hypothesis. When testing merge, a high p-value indicates the merge is good, while for splitting, a low p-value indicates it is good. In a nutshell:

1. If there is a split that results in a p-value less than 0.0500, perform the split with the lowest p-value.

2. If there is a merge that results in a p-value greater than 0.0500, perform the merge with the highest p-value.

3. Otherwise, perform a color operation.

So the merge and split process is performed based on most certain solutions in RTI+. The color operation is applied from red-blue algorithm for state-merging. For more details about merging and splitting process, [5] gives clear demonstration. algorithm 1 is the definition of RTI+ algorithm.

Algorithm 1: RTI+: Real-time identification from positive data

---

Require: A multi-set of timed strings $\{S_+\}$ generated by a PDRTA A_t
Ensure: The result is a small DRTA A, in the limit $A = A_t$
Construct a timed prefix A tree from $\S_+$, color the state q_0 of A red;
while *A contains non-red states* **do**
  Color blue all non-red target states of transitions with red source states;
  Let $\delta = \langle q_r, q_b, a, g \rangle$ be most visited transition from a red to a blue state;
  Evaluate all possible merges of q_b with red states;
  Evaluate all possible splits of δ;
  if *the lowest p-value of a split is less than 0.0500* **then**
    perform this split;
  else
    if *the highest merge p-value is greater than 0.0500* **then**
      perform this merge;
    else
      color q_b red
    end
  end
end

To implement RTI+ on the discrete event sequence, a few steps are needed as preprocessing, which are as followings:

- The first notion is to adjust the data type. RTI+ accepts certain format of the input data, which is demonstrated in Figure 4.5. The first line of data contains the amount of examples/frames and the amount of alphabets appear in the input data, which can be viewed as the amount of sentences in one paper and the words used in it. Then in the following data, each line stands for an example frame, which can be seen as each sentence. In each data line, the first number is the amount of events in this frame, then follows the format of event, time stamp and repeated. The time stamp needs to be adjusted in integer format. Based on our test data, event "0" and "1" have been kept still, while event "A" and "B" have been replaced by 2 and 3 respectively.

```
280 4
28 2 0 2 0 2 0 2 0 2 0 1 124 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0
28 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0
28 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0
28 2 20 2 0 2 1 2 0 2 1 1 275 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 275 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 20 2 0 2 1 2 0 2 1 1 276 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0
28 2 25 2 0 2 1 2 0 2 1 1 271 1 0 3 20 3 0 3 1 3 0 3 1 0 276 0 0 2 25 2 0 2 1 2 0 2 1 1 271 1 0 3 25 3 0 3 1 3 0 3 1 0 271 0 0
28 2 25 2 0 2 1 2 0 2 1 1 271 1 0 3 25 3 0 3 1 3 0 3 1 0 271 0 0 2 25 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0
28 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0
28 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0
28 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0
28 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0 2 24 2 0 2 1 2 0 2 1 1 271 1 0 3 24 3 0 3 1 3 0 3 1 0 271 0 0
```

Figure 4.5: an example of RTI+ input structure, which requires the format of time difference and symbol. The first line is the quantity of frames and the number of symbols. Event "A" and "B" have been replaced by 2 and 3 respectively.

- The second notion is how to segment the frames/to decide the frame length. In another word, how to define the length of a 'sentence' in the discrete event sequence. The segmented frame length cannot be randomly. In our test data, since it already contains a cycling loop, the frame is segmented based on the length of repeated loops. With the consideration of missing loop, it would be better to cut the

frame for two loops, which is from event A to event 0 twice. It is not reasonable to use the fixed length for segmenting frames, which may lead to bias in the learning process. Except the data for learning, the test data also has to be segmented for the automata to read and identify anomaly types.

- The third notion in our research case is the precision scale for time guard. As the description about discrete event sequence above, the precision level of time is still microseconds and the time interval between events is not fixed. Since RTI only accepts time stamps in integer format, the time interval needs to be adjusted with consideration of algorithm accuracy. Here we adjust the time scale with three levels, we set the time stamp to enlarge 10 times, 10^3 times, and 10^6 times respectively, then round the time stamp in integer format. The third setting keeps the precision of time stamp, while the first two has some shifts from the raw time stamp. This step may influence the state-merging process. When merging a state, the p-value has to be higher. However, if the time stamps have big value range, the small differences are magnified, which may possibly lead to a lower p-value. So the state-merging process is failed.
- The fourth notion is the output of the RTI+ algorithm, which needs to be transformed to plot the automata for visualization. The solution is given in the format as following, $0\,2\,[0,25]->1\,\#262\;p=0.9035$ which stands for the starting state, the transition condition/string, time guard/interval, ending state, the amount of examples and the probability of this state transition. The model will be transformed in a visualized method for better understanding the transition and transition condition between events.

4.1.3. GRAMMARVIZ

Consisting of two components, grammarviz[4] has been applied as detection technique for the univariate time series. One component is the representative method of time series, the other one is context-free grammar which analyzes the hierarchy structure after symbolic transformation.

SYMBOLIC AGGREGATE APPROXIMATION

One hot research field, which is also frequently the first step for time series analyzing, is the representative method and transformation of time series data. This is a big assistant for similarity measure in the transformed space. There are multiple methods for representing time series data, generally illustrated in Figure 4.6.The leaf nodes refer to the actual representation, and the internal nodes refer to the classification of the approach. Many methods in Figure 4.6 are widely applied, such as Discrete Fourier Transformation (DFT) [39], Singular Value Decomposition [40], the Discrete Wavelet Transformation [41], and etc. The limitation hidden behind such representative methods is that they are real valued, so the available data structures is narrow. Symbolic representative methods have raised up because of this.

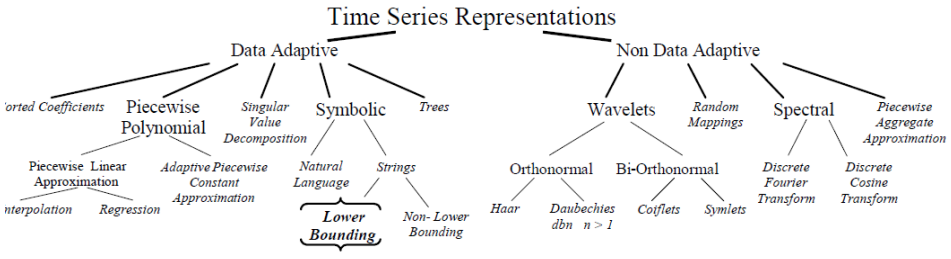


Figure 4.6: A General Hierarchy of Most Various Time Series Representation Methods.[6]

The lower bounding symbolic approach in Figure 4.6 is called Symbolic Aggregate Approximation (SAX), brought by Eamonn Keogh, Jessica Lin, and et al.[6]. SAX is one of the popular time series representation methods. The strong advantage of SAX is that it supports both dimensionality/numerosity reduction and distance measures. The distance measurement is defined on the symbolic approach that lower bound corresponding distance measures defined on the original series, this is improved than other symbolic representation. Generally, SAX contains two steps. The first step is transforming the data into Piecewise Aggregate Approximation (PAA) representation and then symbolizing the representation into a discrete string.

Among the time series representation methods, PAA already has well-defined and well-documented dimensional reduction advantage. To be specific, a time series of C of length n represented in a w-dimensional

[4] http://grammarviz2.github.io/grammarviz2_site/index.html

space by a vector $\overline{C} = \overline{c}_1, ..., \overline{c}_w$. The i^{th} element of $\overline{C}$ is calculated by the equation: $\overline{c}_i = \frac{w}{n} \sum_{j=\frac{w}{n}(i-1)+1}^{\frac{w}{n} \times i} c_j$. To state it, the data is divided into w equal sized "frames" to reduce the time series from n dimensions to w dimensions. The mean value of the data falling within a frame is calculated and a vector of these values becomes the data-reduced presentation. In another word, the representation is an attempt to approximate the original time series with a linear combination of box basis functions. After obtaining this PAA approximation, the PAA coefficients are mapped into SAX symbols based on the predetermined breakpoints. The predetermined breakpoints are determined to produce equal-sized areas under Gaussian curve, with the condition that time series fits Gaussian distribution after normalization. For example, all PAA coefficients below the smallest breakpoint are mapped to the symbol "**a**", all coefficients greater than or equal to the smallest breakpoint but less than the second smallest breakpoint are mapped to the symbol "**b**". The symbols "**a**", "**b**" are approximately equiprobable and the concatenation of symbols that represent a subsequence is a *word*. A subsequence C of length n can be represented as a *word* $\hat{C} = \hat{c_1}, ..., \hat{c_w}$. This is how the SAX works, transforming the time series to readable sequences. Eamonn Keogh, Jessica Lin and other co-workers defines a MINDIST function which returns the minimum distance between the original time series of two words. Given two time series Q and C with the same length n, their Euclidean distance is defined as

$$D(Q,C) = \sqrt{\sum_{i=1}^{n} (q_i - c_i)^2}$$

Transforming the original subsequences into PAA representations, $\overline{Q}$ and $\overline{C}$, a lower bounding approximation of the Euclidean distance between the original subsequences is

$$DR(\overline{Q},\overline{C}) = \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^{w} (\overline{q_i} - \overline{c_i})^2}$$

The MINDIST function is then defined as

$$MINDIST(\hat{Q},\hat{C}) = \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^{w} (dist(\hat{q_i},\hat{c_i}))^2}$$

It is proved that this MINDIST lower bounds the PAA distance. And it turned out that the PAA distance lower bounds the true Euclidean distance between two original time series [42]. Thus, by transit, MINDIST lower bounds the Euclidean distance, which is one advantage of SAX. The demonstration process is with full details in [6]. After transforming the time series with SAX, the time series becomes strings and the second step is to use context-free grammar for analysing the sequences.

CONTEXT-FREE GRAMMAR

Context-free grammar applies for strings without consideration of context. It is a set of recursive rewriting rules or productions used to generate patterns of strings [43]. All the rules are simple replacements starting from one, with mappings to multiple amount items. To generate a string in the language, context-free grammar begins with a string consisting of only a single start symbol. The production rules are applied in random order, until a string that contains neither the start symbol nor designated nonterminal symbols is produced. A production rule is applied to a string by replacing one occurrence of the production rule's left-hand side in the string by that production rule's right-hand side. The language formed by the grammar includes all distinct strings that can follow this manner. For example, assume the alphabet contains a and b, the start symbol is S, and the production rules are as following:

$$1. S \rightarrow aSb$$

$$2. S \rightarrow ba$$

Starting from S, if rule 1 is chosen, then the string aSb is obtained. Applying rule 1 again, S is replaced with aSb, then string aaSbb is obtained. If now apply rule 2, S is replaced by ba and the string aababb is obtained, and done. This process can be written more briefly as: $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aababb$. The language of the grammar is then the infinite set $\{a^n bab^n \mid n \geq 0\} = \{ba, abab, aababb, aaababbb\}$, where a^k is a repeated k times and n represents the number of times production rule 1 has been applied.

Sequitur[44] is one kind of context-free grammars which analyses hierarchical structure in the sequences. Sequitur uses non-terminal symbol to replace repeated subsequence for more concise representation of the

whole sequence. This algorithm has two properties called digram uniqueness and rule utility, which are also constraints when implemented. Digram uniqueness means no pair of adjacent symbols appears more than once in the grammar and rule utility referred to every rule is used more than once. For existing rule S for a whole sequence, a new symbol and its predecessor(the last two symbols) forms the new diagram which should not occurs before, if it appears before, a new rule will replace the original diagrams. The new rule is based on the tail-side digram and headed by a new non-terminal symbol. Actually if the new rule is the same as an existing rule, it will not be created, otherwise it will replace the original diagram.

The combination of SAX and Sequitur has been developed to be an open-source software by Pavel Senin, Jessica Lin, and et al. [45], which can generalize multiple grammar rules and detect anomalies.

4.2. PRINCIPAL COMPONENT ANALYSIS

Principle Component Analysis (PCA) uses a set of linearly uncorrelated variables which are called principal components to represent a set of observations of possibly correlated variables by an orthogonal transformation. The first principal component has the largest possible variance, which means it accounts for the most variability of the original data space. Actually each following component has the highest variance with the limitation of being orthogonal to the preceding components [46]. And it always been used as assistant for multiple techniques to reduce the original data space, because the number of principal components is less than the number of original variables, sometimes even only the first few principal components have been used as a general behavioral capture of the original data space. Before implementing PCA, each attribute of the data needs to be normalized (mean centering) so that each variable becomes normal distribution. This step makes sure PCA can capture the true variance and avoid skewing results due to the differences of mean [47].

Similarly with [47], combined with subspace-based fault detection in multivariate process control [48]. Here we define $\mathbf{Y}$ as a $t \times m$ matrix, in which t stands for the time intervals and m represents the measurement (the attributes in the data set we obtained). Then each column i represents the time series of the i-th attribute and each row j denotes an instance of all attributes at time stamp j. Then we apply PCA on our monitoring data, treating each row as a point in data space. $\mathbf{Y}$ is already normalized to make sure PCA could capture the true variance. In the following explanations, $\mathbf{Y}$ will stand for the mean-centered monitoring data. Applying PCA to $\mathbf{Y}$ yields a set of m principal components, $\{\mathbf{V_i}\}_{i=1}^{m}$. The first principal component $\mathbf{v_1}$ points in the direction of maximum variance in $\mathbf{Y}$:

$$\mathbf{v_1} = \arg\max_{\|\mathbf{v}\|=1} \|\mathbf{Yv}\|$$

where $\|\mathbf{Yv}\|^2$ is proportional to the variance of the data measured along v. Once the first k-1 principal components determined, the k-th principal component corresponds to the maximum variance of the residual. And the residual is the difference between the original data and the data projected onto the first k-1 principal axes. So, the k-th principal component $\mathbf{v_k}$ can be written as

$$\mathbf{v_k} = \arg\max_{\|\mathbf{v}\|=1} \|\left(\mathbf{Y} - \sum_{i=1}^{k-1} \mathbf{Yv_i v_i^T}\right)\mathbf{v}\|$$

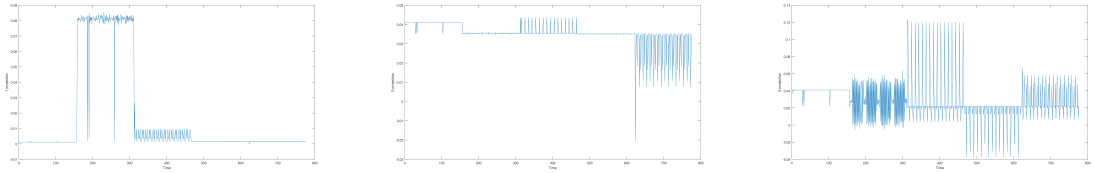
. Through examining the amount of variance captured by principal component, whether the variability in the data can be captured by space of lower dimension is clear.

With the determination of principal axes, the mapping of data to principal axis i is given by $\mathbf{Yv_i}$. Through dividing it by $\|\mathbf{Yv_i}\|$, this vector can be normalized to unit length. So for each principal axis i,

$$\mathbf{u_i} = \frac{\mathbf{Yv_i}}{\|\mathbf{Yv_i}\|}$$

where i=1,2,...,m. The $\mathbf{u_i}$ are vectors of size t and are orthogonal by construction. The above equation shows that all the attribute counts,produces one dimension of the transformed data when weighted by $\mathbf{v_i}$. Along principal axis i, $\mathbf{u_i}$ captures the temporal variation to the entire monitoring time series. Because the principal axes are obtained by order, $\mathbf{u_1}$ captures the largest temporal trend to the entire data set, $\mathbf{u_2}$ captures the next largest, and so on. Figure 4.7 following is the example of original data projection on principal components, iteratively the first principal component capture the most variation, so its pattern is most similar with the original data pattern in Figure 5.14.

The principal axes are separated into two sets, regarding as normal and anomalous variations with the help of PCA. The normal subspace S and the anomalous subspace $\tilde{S}$ span the set of normal and anomalous



(a) $\mathbf{u_1}$ (b) $\mathbf{u_2}$ (c) $\mathbf{u_5}$

Figure 4.7: projections onto principal components example

axes respectively. The theory for detecting and identifying anomalies is drawn from the theory for fault detection in multivariate process based on a subspace [48–50]. The assumption is that the monitoring data $\mathbf{y}$ along time scale can be separated into normal and anomalous components, which refer to *modeled* and *residual* parts of $\mathbf{y}$. Given a point in time $\mathbf{y}$,

$$\mathbf{y} = \hat{\mathbf{y}} + \tilde{\mathbf{y}}$$

,$\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}$ represents modeled and residual monitoring data respectively. We projecting $\mathbf{y}$ onto S and $\tilde{S}$ to form $\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}$. We define a matrix $\mathbf{P}$ of size $m \times r$, consisting of columns which are the set of principal components corresponding to the normal subspace $(\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_r})$, r stands for the number of normal axes. Thus $\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}$ can be written as: $\hat{\mathbf{y}} = \mathbf{P}\mathbf{P}^\mathbf{T}\mathbf{y} = \mathbf{C}\mathbf{y}$ and $\tilde{\mathbf{y}} = \left(\mathbf{I} - \mathbf{P}\mathbf{P}^\mathbf{T}\right)\mathbf{y} = \tilde{\mathbf{C}}\mathbf{y}$ where the matrix $\mathbf{C} = \mathbf{P}\mathbf{P}^\mathbf{T}$ stands for the linear operator which performs projection onto the normal subspace S, and likewise $\tilde{\mathbf{C}}$ onto the anomaly subspace $\tilde{S}$. Thus, $\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}$ contains the modeled and residual monitoring data. A normal instance that satisfies the correlation structure will have a lower value, but an anomalous instance deviating from the structure will have a large value [20]. In another word, anomaly will lead to a large change to $\tilde{\mathbf{y}}$ in general. A common statistic parameter for detecting anomalous change is the squared prediction error (SPE):

$$SPE = \|\tilde{\mathbf{y}}\|^2 = \|\tilde{\mathbf{C}}\mathbf{y}\|^2.$$

For the abnormal changes in residual vector, compare the squared prediction error(SPE,the squared norm of residual value) with σ_α^2, where σ_α^2 donates the threshold for the SPE at the $1-\alpha$ confidence level.

4.3. SIMILARITY BASED DETECTION TECHNIQUES

Similarity measurement is the key issue in time series anomaly detection. Comparatively, the degree of similarity is higher, the less probability of being an anomaly. The similarity measurement influences the justification of anomalies, together with the classification of anomaly scale, which corresponds to the desired output as anomaly label and anomaly score. One of the simplest similarity measures method for time series is Euclidean distance measure, which is also the most widely used distance measure for similarity search [51]. Let x_i and v_j each be a P-dimensional vector, then Euclidean distance is computed as

$$d_E = \sqrt{\sum_{k=1}^{P} (x_{ik} - v_{jk})^2}$$

[52]. In our high dimension data set, each time unit or sequence (same length) can be viewed as an instance in the data space, which is also a vector in Euclidean space. If the distance between two instances is larger, the similarity is lower. There are limitations of Euclidean distance measure for both axis. For Y-axis, though two time sequences may have similar pattern, if they have big difference in amplitude scale, the Euclidean distance will be bigger. Normalization and scaling for amplitude is essential for overcoming the limitation. For time-axis, two time sequences have to keep aligned, otherwise other methods will be better for measure similarity, for example Dynamic Time Warping distance measure. In this research, Euclidean distance measurement is the main similarity measurement method combined with k-means clustering and Nearest Neighbour based techniques. Considering the restriction rooted from the algorithm, the data set trained in this section is also after preprocessed so that the multivariate time series is continuously with same sampling time for each attribute.

4.3.1. k-MEANS CLUSTERING

As the main strategy for unsupervised learning, clustering is the process of grouping similar objects/records into subsets/clusters [53]. It involves the core idea of comparing and measuring similarity, which is also the

core of detecting anomalies in time series data stream. k-means clustering is the one of the most popular clustering algorithms used in scientific and industrial applications [54]. And in practice it has appealing simplicity and speed. It can be used to classify the input data set into k clusters and can also be the input representation as the first feature learning step. It partition n observations into k sets ($k \leq n$) for minimizing a sum squares cost function, which means to minimize the following value:

$$\sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2$$

where μ_i is the mean of points in $\S_i$. k-means clustering has some varied techniques, the k-means clustering implemented in this research is called Lloyd-Forgy [55], which is also the standard algorithm. The standard algorithm steps are as followings:

1. Place k points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the k centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

In another word, the standard algorithm consists of two main steps, one is the assignment and the other one is the update step. The assignment step is to assign each observation to the cluster whose mean yields the least within-cluster sum of squares.[56] From mathematical scope, this means partitioning the observations according to the Voronoi diagram generated by the means[57]. The update step is to calculated the new means to be the centroids of the observations in the new clusters. The initialization method used in this research is Forgy method. It uses the initial means from several randomly chosen observations and assign the rest to the nearest representing cluster.

As a heuristic algorithm, there is no guarantee that the k-means clustering will converge to the optimum. Besides, the initial setting of centroids is an influential factor for the clustering result, both its amount and initialization of its location. k-means clustering has been implemented in order to get a general understanding of features, considering each time unit as the comparison scale. The input data set without unusual patterns have been trained with different number of clusters. Each time unit stands for each data point, the maximum Euclidean distance between time unit and corresponding centroid in each cluster group is calculated. For an unknown testing data set, the distance among each data point and the known centroids has been computed. If the data point owns the distance larger than radius of each cluster which means it cannot be grouped to any known feature sets, it should be an anomaly time unit.

4.3.2. K-NEAREST NEIGHBOUR

Nearest Neighbour search contains variant models, such as the famous k-Nearest Neighbors (k-NN) algorithm, which is to define the top k nearest neighbours according to the query. The basic concept of Nearest Neighbour is to find the 'closest'(or most similar) points according to a query point, which directly fits the similarity comparison for anomaly detection in the data feature space. As a non-parametric technique, the advantage about k-NN is that it does not requite a priori assumption about the data structure, but aims at building the data structure directly [58].

In this research, k-NN is not used for classification or regression, but for the similarity measurement. The similarity between data points is also measured by Euclidean distance. Compared with k-means clustering, the advantage of using k-NN for similarity measurement is that it avoids the bias rooted from the algorithm itself, but gives the direct representation of the data space structure from similarity scope. The closer between data points, the higher similarity they are. For multivariate data instances, distance or similarity is usually computed for each attribute [59], while in this research, it has been used for the similarity comparison between two data sets. In practical implementation, two same time interval data sets have been trained, one is the normal data set, the other one contains the meta information delay problem. The query point is each data point itself and the Euclidean distance for its nearest k neighbours have been calculated. In order to check which parameter is better for measuring similarity, the sum, maximum, minimum and average distance of each point with its neighbours have been calculated. Also, the k has been set differently for testing.

5

Results Analysis

This chapter discusses the details of the experimental process and compares the test results of applied anomaly detection techniques based on the given data sets. In this research, multiple techniques have been proposed for three kinds of data sets. For each technique, varied preprocessing steps are taken for implementation. The goal is to test if each technique to find, locate and even classify the anomalies. And the comparison of each technique is within group, which means the performance of techniques applied on same data kinds will be compared.

5.1. Discrete Event Sequence

As discussed in Chapter 4, two kinds of grammatical inference techniques have been implemented for discrete event sequence, one is based on N-gram, the other one is RTI+ algorithm [24]. The results of two algorithms are given in the following sections. Figure 5.1 is the brief example of normal data set which is applied in the test, the other two are the examples containing time delay and information loss error respectively.

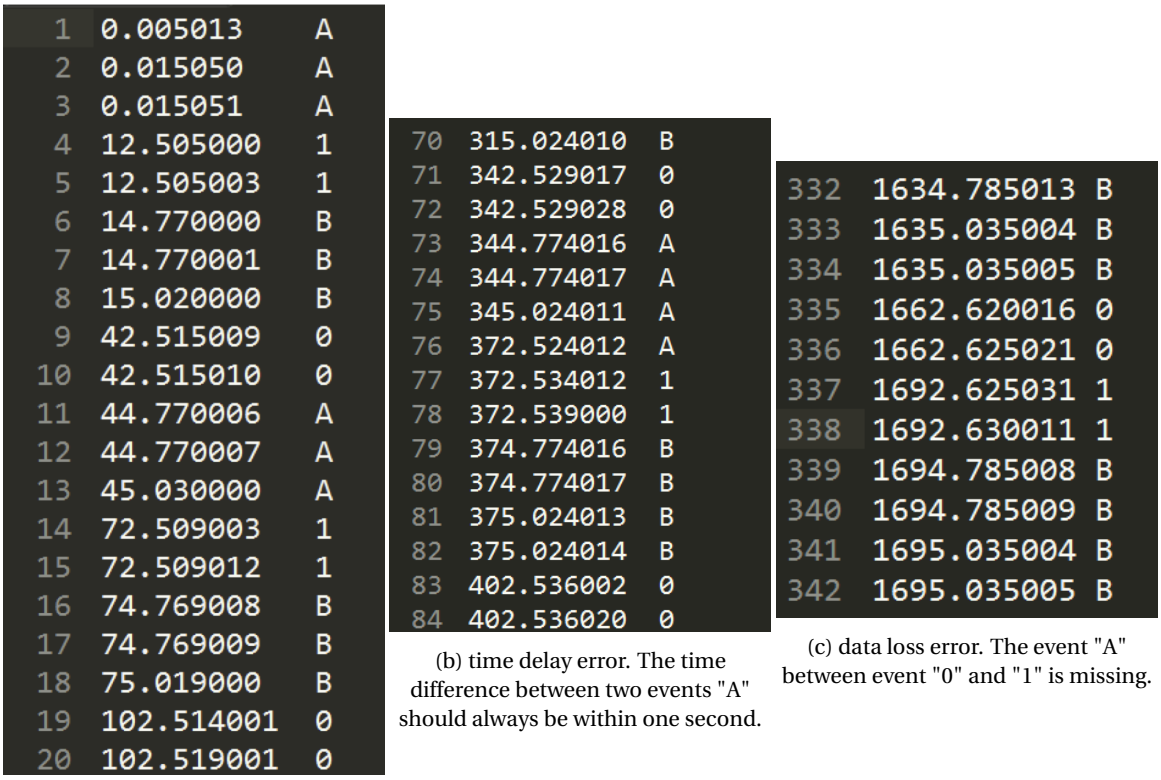
5.1.1. N-gram

Experimental Process

As discussed in the earlier Chapter, in order to consider time constraints, empty values have been filled to obtain equal sampling time. When using N-gram for anomaly detection, the simple way is to compare the probability of event transitions in the data sets. The normal data set should be trained to obtain the standard result, if the test data has the same result, then the test data contains none anomalies. During the experimental process, two data sets have been compared, one contains data lost information anomaly, the other one is the normal data set. The two data sets applied have the same quantity of events in one loop before inserting empty values, with the same length of 3375 indexes after adjustment.

N-gram Result

The result of N-gram is explained as following Table 5.1, alphabet size equals four, since there are four events in total. For each n, the upper row is the result of the normal data set and the lower row contains result of the abnormal data set. The probability is the corresponding event transitions, which means only existing event transitions are included (zero probability has been excluded). As discussion in the experimental process, once the amount of event transition probability have same amount and same value, with high consistency, the test data has no errors. The probability value is limited to 10^{-2}. Apparently, with the increasing of n, the difference between normal and abnormal probabilities becomes more obvious. The amount in the second in the table is the amount of probabilities. When n is settled down, the amount of existing event transition is limited. For example, when $n = 2$, there are eight event transition cases, from "A" to "A", from "A" to "1", from "1" to "1", from "1" to "B", from "B" to "B", from "B" to "0", from "0" to "0", from "0" to "A". This amount should be the same shared by both test data sets. And the amount of probability changes because when n is changing, the gram structure becomes different. The 0.00 value in the table exists because the result is round to 10^{-2}. However, this value of normal data set is different from the one of abnormal data set, which means the probability distribution has changes. The advantage to apply N-gram for discrete event sequence



1	0.005013	A
2	0.015050	A
3	0.015051	A
4	12.505000	1
5	12.505003	1
6	14.770000	B
7	14.770001	B
8	15.020000	B
9	42.515009	0
10	42.515010	0
11	44.770006	A
12	44.770007	A
13	45.030000	A
14	72.509003	1
15	72.509012	1
16	74.769008	B
17	74.769009	B
18	75.019000	B
19	102.514001	0
20	102.519001	0

(a) example of normal discrete event sequence

70	315.024010	B
71	342.529017	0
72	342.529028	0
73	344.774016	A
74	344.774017	A
75	345.024011	A
76	372.524012	A
77	372.534012	1
78	372.539000	1
79	374.774016	B
80	374.774017	B
81	375.024013	B
82	375.024014	B
83	402.536002	0
84	402.536020	0

(b) time delay error. The time difference between two events "A" should always be within one second.

332	1634.785013	B
333	1635.035004	B
334	1635.035005	B
335	1662.620016	0
336	1662.625021	0
337	1692.625031	1
338	1692.630011	1
339	1694.785008	B
340	1694.785009	B
341	1695.035004	B
342	1695.035005	B

(c) data loss error. The event "A" between event "0" and "1" is missing.

Figure 5.1: normal and anomalous data sets

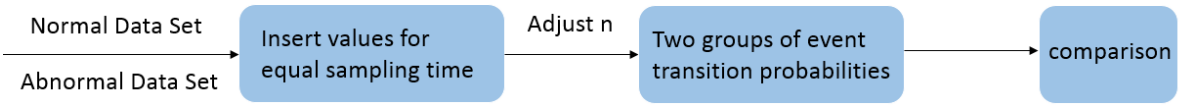


Figure 5.2: Experimental process based on N-gram

is that it is simple to get the standard for justifying anomaly and fast speed. The disadvantage is that it adjust the time in the original data set. Once the whole loop is missing, it would be hard to detect because it does not influence the transition probability. The result of same n are corresponding to the same circumstance. The time delay anomaly does not add unexpected event transition probability, but it influences the probability distribution. In other words, when time delay happens, the amount of one kind event transition increases, which influences the probability distribution in the data set. The event delay happens between two "A" events, which adds the probability of event transition "A" to "A" in the anomalous result.

The false positive of N-gram is zero, when it detects anomalies of timing delay anomaly. However, there is restriction of the event loop. This restriction also exists for detecting data lost error. Furthermore, the lost data cannot be an entire event loop. N-gram cannot detect this whole event loop lost successfully because the probability of event transition is not influenced. Also, N-gram cannot handle the timing error anomaly, unless the timing issue causes consequences of missing data. Though it detects specific anomaly successfully with zero false positive, it cannot give indication about where happens. To further locate the anomaly in the data set, other algorithms should become assistant. N-gram is more like a roughly identification of whether the data set contains losing data (not an entire loop) anomaly. In general, N-gram has a good performance, but it has too many restrictions, which makes it could be easily replaced by other better performed algorithms.

n	amount	probabilities of event transition
n=2	6	0.50, 0.50, 1.00, 0.07, 0.96, 0.99
	6	0.50, 0.50, 0.99, 0.07, 0.96, 0.01
n=3	5	0.96, 1.00, 0.04, 0.04, 0.96
	7	0.96, 1.00, 0.04, 0.04, 0.96, 0.04, 0.02
n=4	5	1.00, 0.96, 0.96, 0.04, 0.04
	6	1.00, 0.96, 0.96, 0.02, 0.04, 0.04
n=5	5	1.00, 0.04, 0.96, 0.04, 0.96
	9	1.00, 0.96, 0.02, 0.02, 0.04, 0.96, 0.04, 0.00, 0.04
n=6	8	1.00, 0.04, 0.96, 0.96, 0.04, 0.98, 0.04, 0.96
	10	1.0, 0.02, 0.96, 0.96, 0.04, 0.04, 0.04, 0.00, 0.98, 0.04
n=7	8	0.95, 1.00, 0.04, 0.96, 0.04, 0.04, 0.96, 0.04
	12	0.00, 1.00, 0.00, 0.02, 0.04, 0.02, 0.98, 0.96, 0.04, 0.96, 0.04, 0.04

Table 5.1: N-gram Result.
For every n, the upper row is the result for the normal data and the lower one is for the abnormal data. Column amount represents the amount of existing event transition in the data set. And column probabilities are the probabilities of event transition, which are round at the scale of 10^{-2}.

5.1.2. RTI+

EXPERIMENTAL PROCESS

As discussed in the earlier Chapter, the input for RTI+ is the discrete event sequence after adjustment of format and sampling time scale. The normal data sets have been segmented into 290 frames with 4 alphabets as the input for RTI+. The training result is desired automaton, which has to read the test data set for performance measurement. The test result is the rejection and classification of anomalies. Two test data sets have been read by the automata, one contains the error of time delay and the other one contains the error of data lost. Both test data sets have been adjusted and cut into frames before putting in the automata.

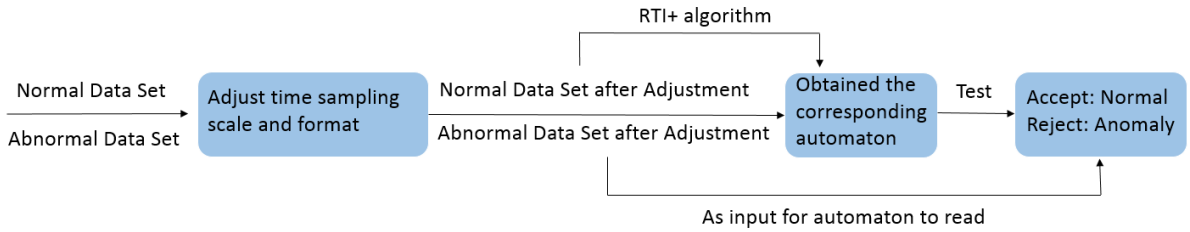


Figure 5.3: time series plots of original and state vector

MODEL INTERPRETATION

Based on the automaton obtained from the normal test data, it is asked to read the test data which contains anomalies. Once it rejects the frame, which means it contains anomaly that is out of the automaton loop. Also, the anomalous test data should be segmented and adjusted in the same format as the normal test data. The automata learnt are shown as following Figure 5.4, Figure 5.5, and Figure 5.6, which ordered by the setting of time scale enlarge 10,10^3,and 10^6 times.

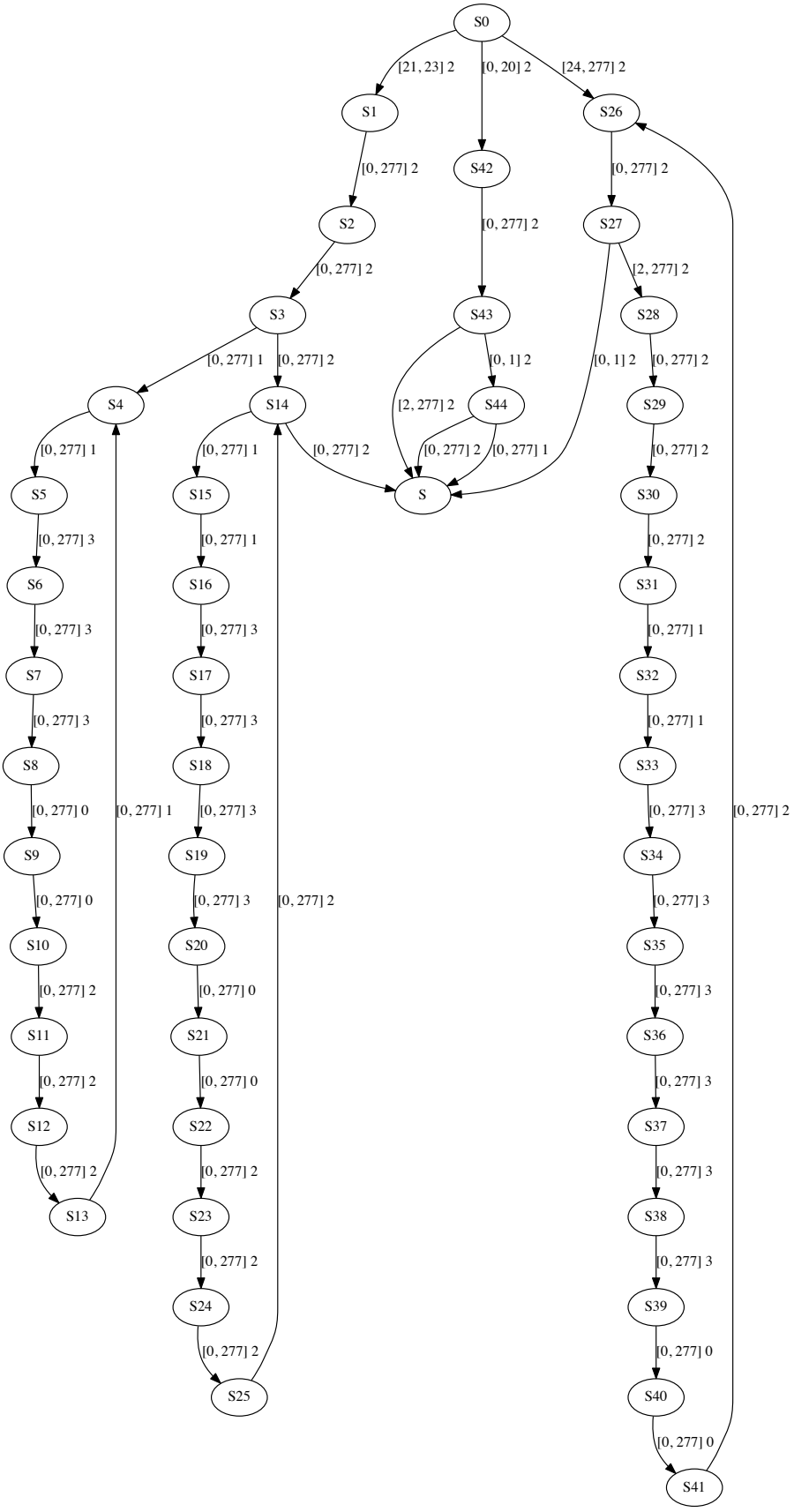


Figure 5.4: Experimental process based on RTI+

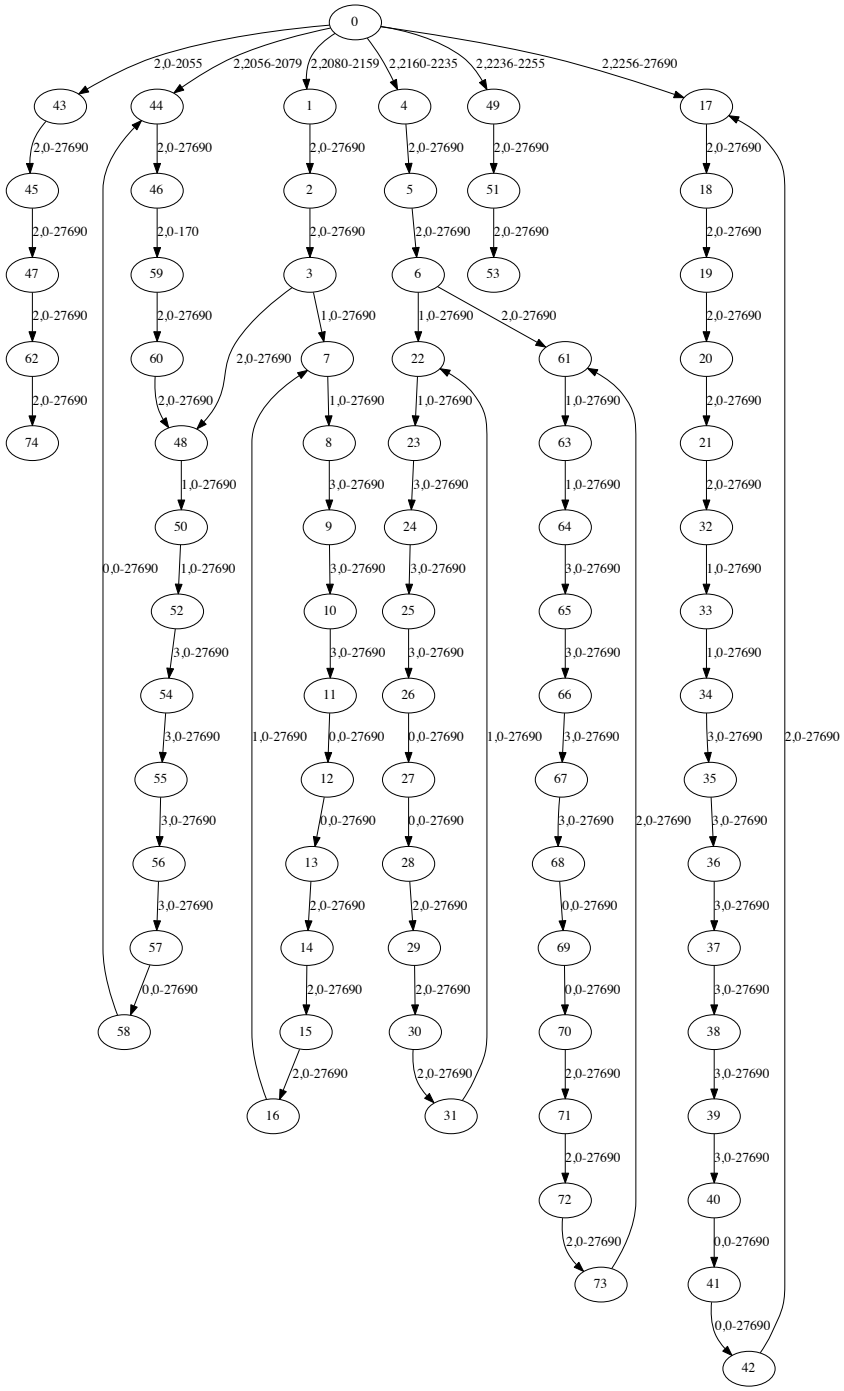


Figure 5.5: the automaton model with time sample enlarge 10^3 times

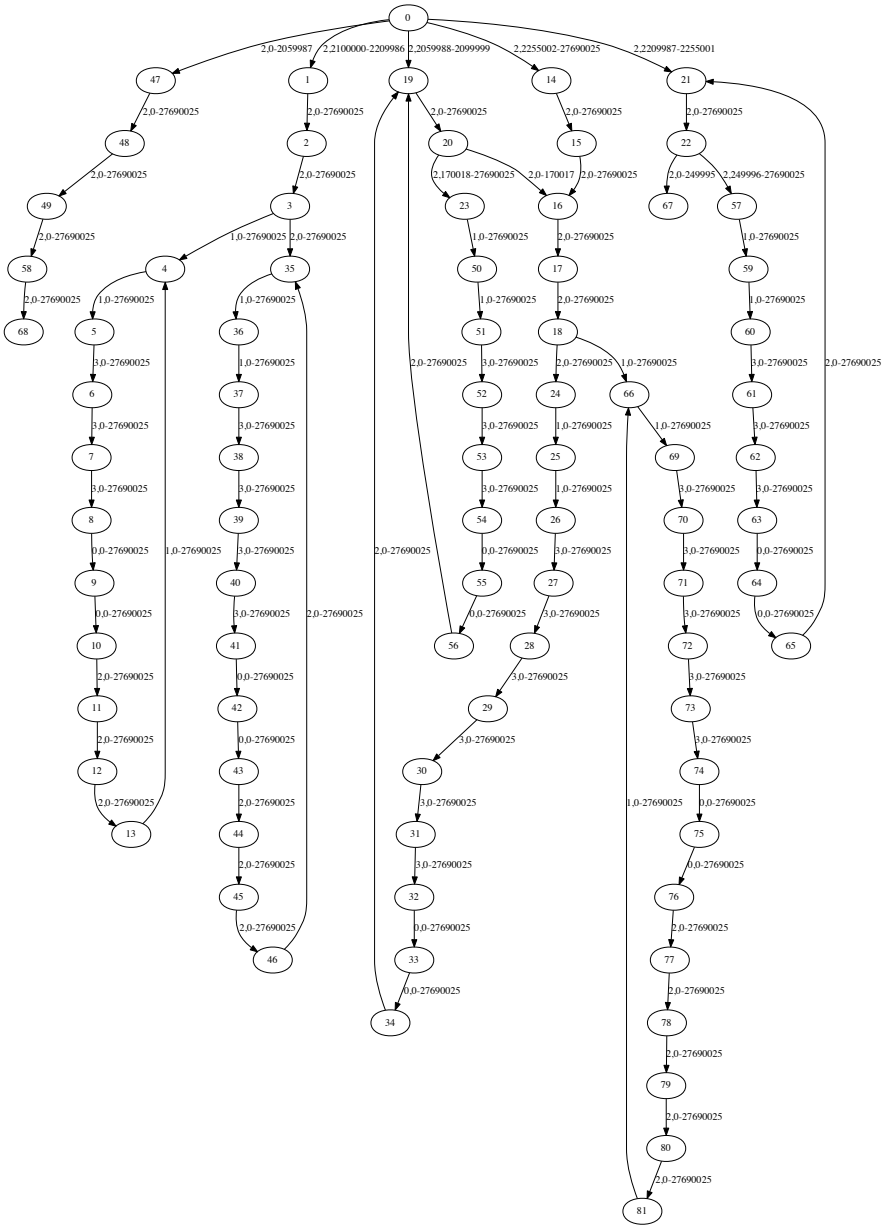


Figure 5.6: the automaton model with time sample enlarge 10^6 times

In the automaton, the state is remarked in circle starting with 's', and the transition condition is remarked as the delay guard and event. The event transitions (data frames during the test process) which does not follow the rules of loop will be rejected as an anomaly. Among the three settings of time precision, 10 times performs best, take its model Figure 5.4 as an example for a detailed explanation. In this model, there are two main components, state and condition. State is marked in the circle, and the condition for state transition is written over the arrow. The state transition condition consists of two elements, one is symbol, the other one is delay guard. Based on our test data, there are four symbols, "0","1","2","3" which stand for event "0","1","A","B" from the training sequence respectively. When both symbol is correct and happening time falls in the delay guard, the state will transmit. The starting state is from the top as $S0$. From $S0$, it is possible to move to three other states, which are $S1$, $S42$, $S26$. And the conditions are different among these three states, though they all limit the alphabet to be "2", the delay guard are different. In another word, event "2" happens within the period of delay guard, the state will move. If the event is not event "2" or the happening time from initial state $S0$ does not follow any of the three delay guards, the data will be rejected and stopped as anomaly. Take a loop on the left side for instance. Event "2" appears within time 2.1 to 2.3 seconds, state $S0$ transfers to state $S1$. From state $S1$ to state $S2$ and state $S2$ to state $S3$, the transferring condition are the same, which is event "2" appears within 27.7 seconds. From state $S3$, there are two possible complete state transition loops. From state $S3$ to state $S4$, then follows a complete loop back to state $S4$. From state $S4$ to state $S13$, the event condition follows the loop as $1 \rightarrow 1 \rightarrow 3 \rightarrow 3 \rightarrow 3 \rightarrow 0 \rightarrow 0 \rightarrow 2 \rightarrow 2 \rightarrow 2$ and then goes back to state $S4$. In another word, after state $S13$, the next state should be $S5$ as long as event "1" appears within 27.7 seconds. The event loop generated from this left side model is $2 \rightarrow 2 \rightarrow 2 \rightarrow 1 \rightarrow 1 \rightarrow 3 \rightarrow 3 \rightarrow 3 \rightarrow 0 \rightarrow 0$ and then repeated this circle. These circling loops capture the existing events' behavior in the training data sets. The model represents the normal events' behavior, while the behavior which does not follow any loop will be identified as an anomaly. There is one thing to be noticed is that there is one ending state S which is different from all the rest loops. State S means that kind of event loop stops there without anomalies. Actually, the existing event loops from $S0$ to S are normal, but they happened with rare probability. When applying this model for anomaly detection, both the event and the time should satisfy the condition over the arrow between states. Except the routine ends in ending state S, all the other routines become cycling loop, indicate the event loop behavior in the training data.

RESULT ANALYSIS

To detect anomalies, let the model read the test data, which is also prepared for the requirement of RTI+. Once there is an anomaly inside the event loop, that piece of data frame will be rejected and classified as either time or symbol anomaly. False positive warning exists because rare normal frames are treated as terminal state during the training process, which refers to state S as discussed before. This false positive warning raised up from the generalization of state merge. The test results are in the format of the rejected data frames, followed the classification of anomaly and counting amount of anomalies in this test data. The symbols within [] is the rejected frame, follows the pattern as $['event','time']$, the time is not the appearance point, but the delay from this event to the previous event. There are two kinds of anomaly could be detected, one is the symbol error, the other one is time, which refers to event and time anomaly respectively. Followed the rejected frame, which is the result of detecting missing anomalies based on 10 times setting of the time scale.

The test result with time scale setting enlarges 10 **times**

['2','0','2','0','2','0','2','0','1','125','1','0','3','23','3','0','3','2','3','0','0','275','0','0','2','23','2',
'0','2','3','2','0','1','275','1','0','3','23','3','0','3','2','3','0','0','275','0','0'] time 1

['1','300','1','0','3','22','3','0','3','2','3','0','0','276','0','0','2','22','2','0','2','2','2','0','1','276','1',
'0','3','22','3','0','3','2','3','0','0','276','0','0'] symbol 15

There are two data sets containing two kinds of anomalies respectively and they are applied for testing the performance of the three automata. One is the anomaly of information lost, which is missing four event "A" in one loop, the other one is the timing delay problem. Considering the anomaly of information lost, since it is not a whole event loop lost, it should be rejected as symbol anomaly during the test process. Among the three automata, the test result which performs the best, is the setting of time scale is round time stamp after increasing 10 times. It explains the existing loop of normal behavior in the system. One case of false positive exists in the 10 times setting, which is the classification of time anomaly. This data frame is normal, while inside it exists the terminate state. In another word, this is normal but happens rarely. However, for the timing delay error, the model with sampling time enlarges 10^6 times performs best with low false positive. The

numeric result of anomalies detected based on the three automata is explained in Table 5.2. The quantity of missing anomaly and timing error is 1 in each abnormal data set, and the values in the table are the quantity of anomalies detected by RTI+. During the testing process, each discrete event is treated as one data instance,

Settings	Missing Error(1)	Timing Error(1)
10 times	2	29
10^3 times	29	29
10^6 times	29	13

Table 5.2: Test Results of Three Automata.
There is one missing error, and one timing error in each testing data set. The number in the table stand for the anomalies detected.

the model performance is evaluated according to the confusion matrix Table 5.3. The predicted condition is the testing result obtained from the automata and the true condition reflects the situation in the test data set. Corresponding rates are also computed as measurement for the automata performance, which are True Positive Rate (TPR), False Negative Rate (FNR), False Positive Rate (FPR) and. TPR demonstrates the probability of detection and the sensitivity of detection technique, while FPR demonstrates the probability of false alarm. The Table 5.4 and Table 5.5 represent performance measurement of the automata respectively. For data lost

		Predicted Condition	
		Positive	Negative
True Condition	Positive	True Positive(TP)	False Negative(FN)
	Negative	False Positive(FP)	True Negative(TN)

(a) confusion matrix

$$\text{True Positive Rate (TPR)} = \frac{TP}{Condition\ \ Positive}$$
$$\text{False Negative Rate (FNR)} = \frac{FN}{Condition\ \ Positive}$$
$$\text{False Positive Rate (FPR)} = \frac{FP}{Condition\ \ Negative}$$
$$\text{True Negative Rate (TNR)} = \frac{TN}{Condition\ \ Negative}$$
$$\text{Accuracy(Acc)} = \frac{TP+TN}{Total\ \ Population}$$

(b) measurement standards, condition here demonstrates true condition

Table 5.3: confusion matrix and measurements

test data set, there is one anomaly and the quantity of complete data set is 678. For time delay data set, there is also one anomaly and the quantity of the complete data set is 682. The false positive rate becomes higher because more normal instances are identified as anomalies. It happens with the changing of time scale. And false positive cases are mainly classified as timing problem. They are not influenced by the true anomaly, but most of them cannot satisfy the requirement of time guards. Few cases are identified as anomaly because they happen rarely, which leads to the ending state.

	TP	FN	FP	TN	TPR	FNR	FPR	TNR	ACC
10	1	0	1	676	1.00	0.00	0.00	1.00	1.00
10^3	1	0	28	649	1.00	0.00	0.04	0.96	0.96
10^6	1	0	28	649	1.00	0.00	0.04	0.96	0.96

Table 5.4: the result of three automata when testing data lost anomaly

In conclusion, when detecting the anomaly of data lost, the automaton with time sampling 10 times larger performs the best. While when detecting the anomaly of time delay, the automaton with time sampling 10^6 times bigger performs the best. This conclusion is not obtained only from the accuracy, with so few positive examples in both data sets, the accuracy tends to be higher. It is of great importance that all the false negative rates are zero, which means there is no anomaly identified as normal instance. When detecting anomalies, the false positive appear mainly because of the time guards. Some events' loop are normal but they are rejected as anomaly because of small difference in time guards.

	TP	FN	FP	TN	TPR	FNR	FPR	TNR	ACC
10	1	0	28	653	1.00	0.00	0.04	0.96	0.96
10^3	1	0	28	653	1.00	0.00	0.04	0.96	0.96
10^6	1	0	12	669	1.00	0.00	0.02	0.98	0.98

Table 5.5: the result of three automata when testing time delay anomaly

In general, for discrete event sequence, both N-gram and RTI+ have limitations from the practical scope. Though preprocessing step of data invovled in N-gram is easier than the step in RTI+, N-gram has more restrictions of the anomalies type and it can only demonstrate whether there is an anomaly in the data set. Comparatively, RTI+ has more capability of detecting targeted anomalies and even classifying the anomaly types. Also, RTI+ not only give the identification of anomaly but also its location, which saves time and helps the researcher locate the anomaly efficiently. The only consideration with RTI+ is the time setting, which should also be appropriate for more anomalies in the future.

5.2. UNIVARIATE TIME SERIES

5.2.1. GRAMMARVIZ EXPERIMENTAL PROCESS

The methodology of Grammarviz consists of SAX and context-free grammar. When applying it, the only preprocessing step to deal with the data is to transform the univariate time series into a continuous sequence of values, which has same time interval. There are two test data sets, one contains the error of value crossing the limitation and the other one contains the error of data lost. The normal data set is not involved for training because the data sets contain both normal and anomalous situation, where normal time instances are the majority. Figure 5.8 demonstrates the process of applying Grammarviz over the univariate time series.

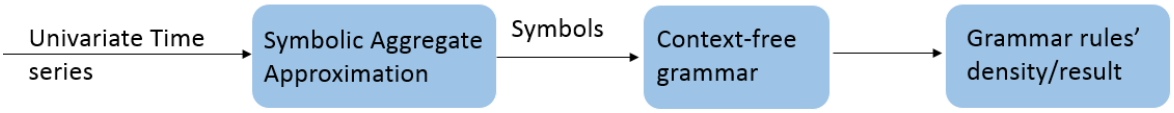


Figure 5.7: Experimental process based on SAX and Context-free grammar

5.2.2. GRAMMARVIZ RESULT

In this research, grammarviz has been applied directly for detecting anomalies in the univariate time series. Figure 5.8 is an example of the user interface and the data set which contains the anomaly of information lost. The first line in the control panel is the loading data, where the data has to be single sequences with same time interval, which is classical univariate time series. The next line in the control panel of SAX parameters, including the window size, the PAA size, and the alphabet size. The setting is more complexed, the time computed is longer. In the right corner of the panel, the anomaly has been demonstrated clearly. All the sudden changes of range value is listed in the anomalies ranking. This result is not persuasive because it is connected with both normal and anomalous time period, which is not collected continuously. But it gives hints about the sudden impulse during the normal time period and how this method works based on the history.

Figure 5.9 gives the hint about anomalies based on the rule density. The blue color behind the data points indicates the density of multiple appropriate grammar rules. The darker the color is, the higher the rule density is. In other words, at this point, the data value is covered by more grammar rules if the blue color is darker, which means it is normal because it is accepted by more context-free grammar rules. In this test, the grammar rules have been pruned for obtaining more accurate result. For example, the white space around index 33000 has been viewed as anomalous part. And the serious anomaly of information lost is indicated by its starting point, where the value suddenly drops to zero.

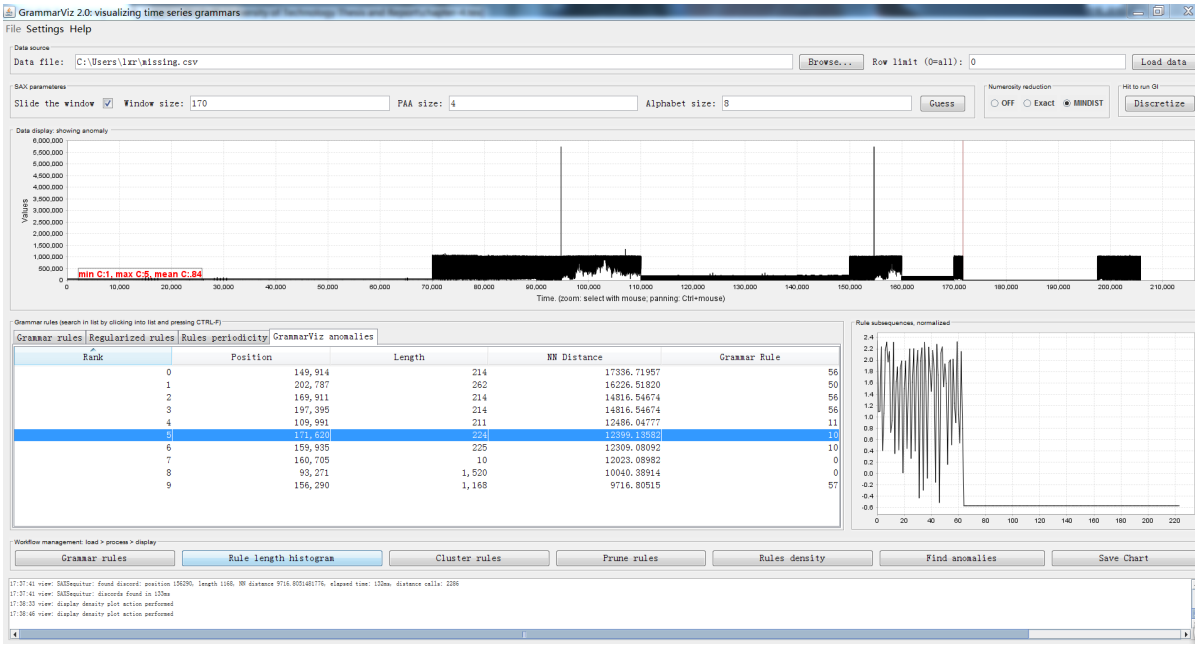


Figure 5.8: the user interface of grammarviz and mixed data set loaded

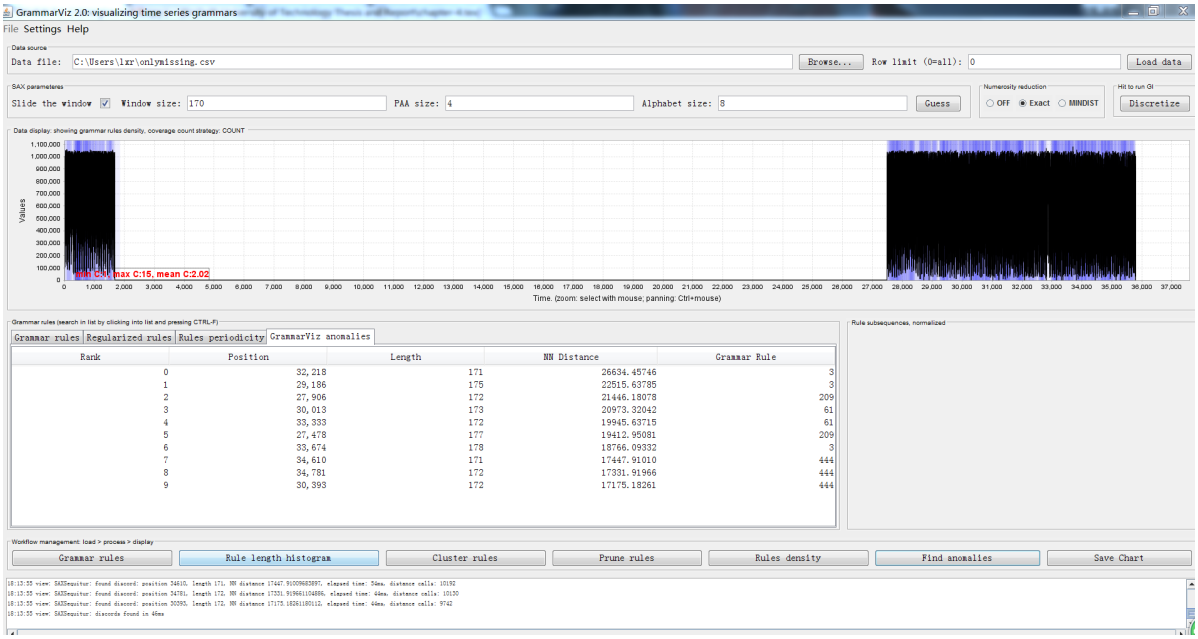


Figure 5.9: the Test Result of Information Lost Anomaly. Usually, the blue color behinds black value indicates the density of possible grammar rules, which is also the anomalous extent of the anomaly. The lighter the blue is, the more anomalous the data instance is. However, though it is totally white in the anomaly period, the ranking of suspicious anomalies has no records of this anomaly. In other words, Grammarviz did not detect this zero value period as anomaly, which may be because it is based on limited history. The scale of the anomaly probably exceeds its capability.

Similarly, Figure 5.10 demonstrates the test result of crossing limitation anomaly. Apparently, with consideration of global understanding about the patterns, this method can find very meticulous changes in the patterns hidden in the univariate time series. If you want to dig more about the normalization and patterns, the details can also be viewed through simple visualization, for example, in Figure 5.11. The suspicious anomalies are marked with green circle, but needs to be further verified. It can find more hidden patterns behind the data set and indicate the severity by color, but there is certain false positive rates. And this is demonstrated in both test results. Another disadvantage is that it cannot handle huge data set efficiently. When large amount of data are loaded in, it will get stuck.

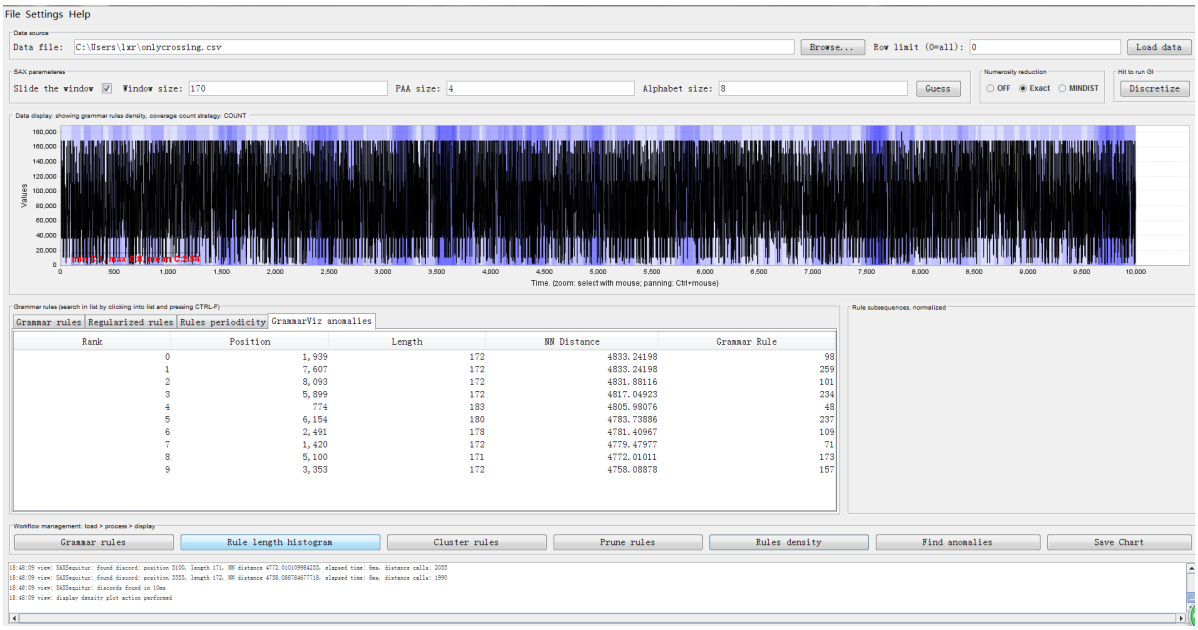


Figure 5.10: the test result of crossing limit anomaly

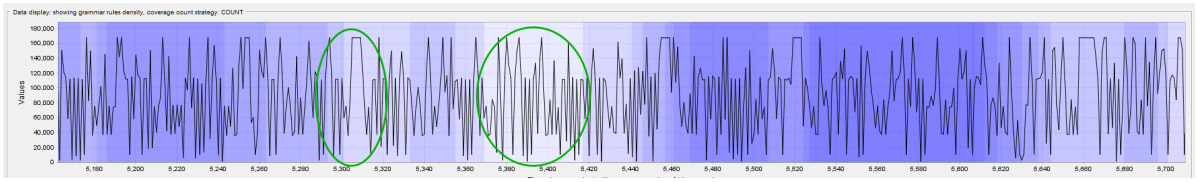


Figure 5.11: the Detailed Figure of Test Result. The test of this anomaly is a black box test, anomaly of crossing limitation happens, but we do not know when it happens. Based on the background knowledge about the system, the suspicious anomalies could be the patterns which keeps highest value steadily, such as the horizontal line around index 5310. In the test result, there are a few suspicious anomalies. The top one is around index 5370 and lasts until index 5410. In general, this test result is not persuasive enough, which needs to be further verified.

5.3. MULTIVARIATE TIME SERIES

5.3.1. PCA

EXPERIMENTAL PROCESS

In this research, a preprocessing step is applied over the multivariate time series before detecting anomalies. To apply PCA for anomaly detection, the process starts with finding the appropriate amount of principle components. And then the original multivariate time series is separated into normal and residual state according to the subspace construction. In this research, the common test method Q-test [48] has been applied as the threshold for detecting anomalies in the residual vector. To implement PCA for anomaly detection, it

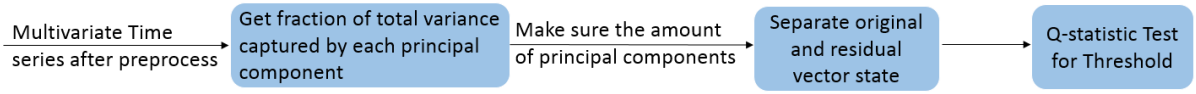


Figure 5.12: Experimental process based on PCA

is important to find the appropriate amount of principle components. The following Figure 5.13 presents the fraction of total variance captured by each principal component of the input. It is not hard to see after 40, the percentage drops below 10 percentage, which means the dimension amount of the training set can be reduced to the first 40 principle components because they could capture nearly the entire data set. Though 30 principle components is also appropriate as it is below 10%, 40 is set up to capture the data set as much as possible.

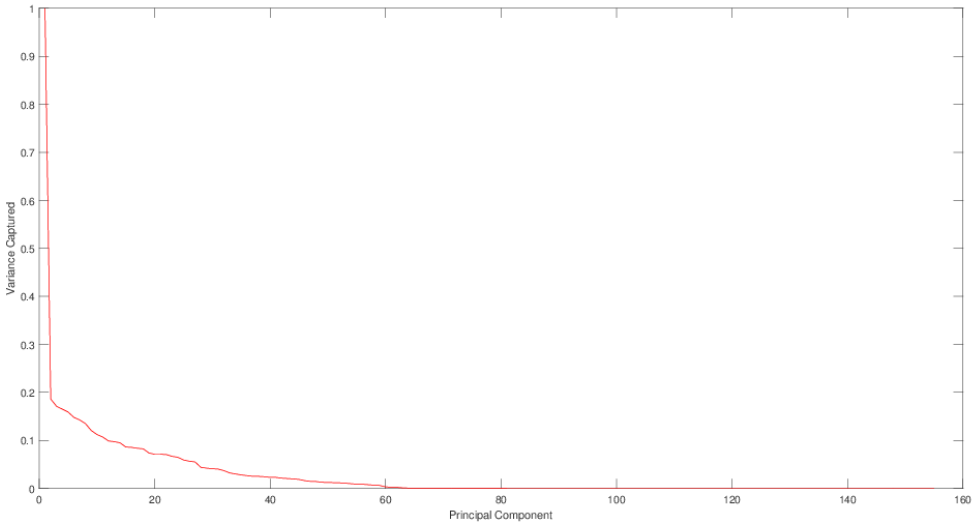


Figure 5.13: fraction of total variance captured by each principal component

Based on an amount of 40 principle components, the original training data set vector state and the norm state vector are demonstrated in Figure 5.14.

The residual vector is in Figure 5.15, which demonstrates clearly the difference from the normal state vector. The surprisingly impulse around 150, 180 and 600 along time scale apparently could be identified as anomalies. Compared with the state vector, the pattern from 150 to 300 in residual vector may become false positive identification because it may contribute to the pattern in the normal state vector, but the impulse around 600 can be identified as an anomaly. The impulse around 600 is so obvious that it only appears in the residual state vector rather than the normal state vector, which means it does not contribute to or be similar with the pattern in the normal state.

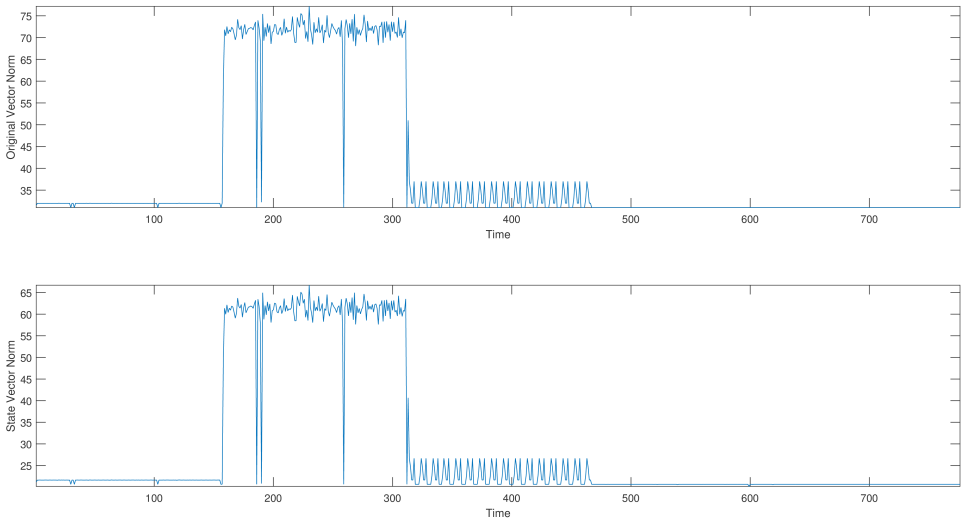


Figure 5.14: time series plots of original and state vector

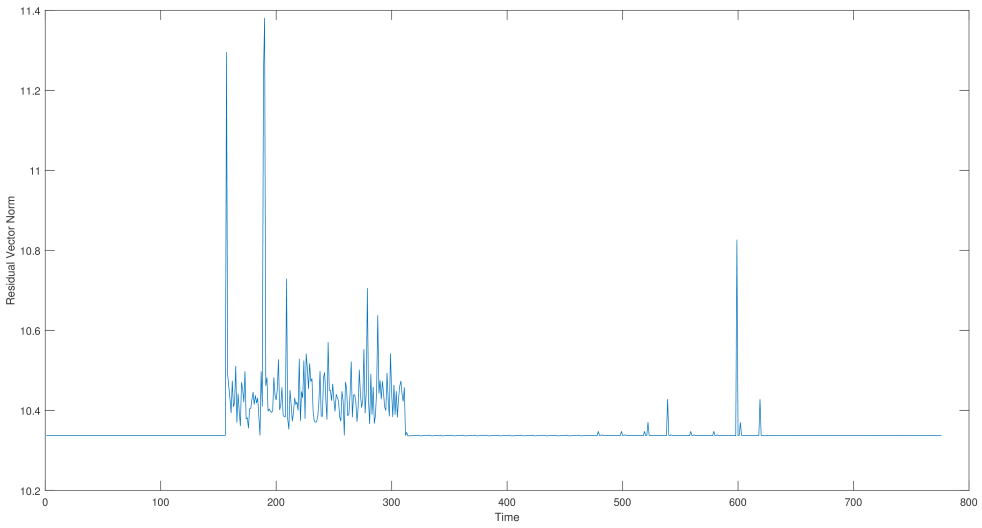


Figure 5.15: time series plots of residual vector

PCA RESULT

Observations from plots give some indications about patterns in the data set, while it is not as persuasive as scientific threshold. Here the threshold is calculated based on the Q-statistic. The Q-statistic for residual vector developed by Jackson and Mudholkar is implemented [48] commonly, with the advantage that its result holds regardless of how many principal components are retained in the norm subspace. In general, if SPE $\leqslant \sigma_\alpha^2$, which means it should be an abnormal change, and σ_α^2 denotes the threshold for SPE at the $1-\alpha$ confidence level. According to Jackson and Mudholkar, σ_α^2 as the threshold compared with SPE is defined as:

$$\sigma_\alpha^2 = \phi_1 \left[\frac{c_\alpha \sqrt{2\phi_2 h_0^2}}{\phi_1} + 1 + \frac{\phi_2 h_0 (h_0 - 1)}{\phi_1^2} \right]^{\frac{1}{h_0}}$$

where $h_0 = 1 - \frac{2\phi_1\phi_3}{3\phi_2}$, and $\phi_i = \sum_{j=r+1}^{m} \lambda_j^i$ for i = 1,2,3. And r is the amount of principal components and m is the amount of time series. λ_j^i here stands for the variance captured by projecting the data on the $j-th$ principal component($\|\mathbf{Yv_j}\|^2$) and c_α is the $1-\alpha$ percentile in a standard normal distribution. In this setting, the $1-\alpha$ confidence limits to the false alarm rate of α, if the assumptions are satisfied before achieving result. The assumption for deriving the confidence limit in Q-statistic is that the sample vector **y** follows a multivariate Gaussian distribution. However, Jensen and Solomon point out when the underlying distribution of the original data substantially from Gaussian distribution, the Q-statistic changes little [60]. However, in our case, the σ_α^2 has a tiny magnitude around 3×10^{-15} that it can not be used as a threshold for abnormal changes. As Figure 5.16 displays, the red dotted line stands for the lower bound of the threshold, which is too low to be an identification scale.

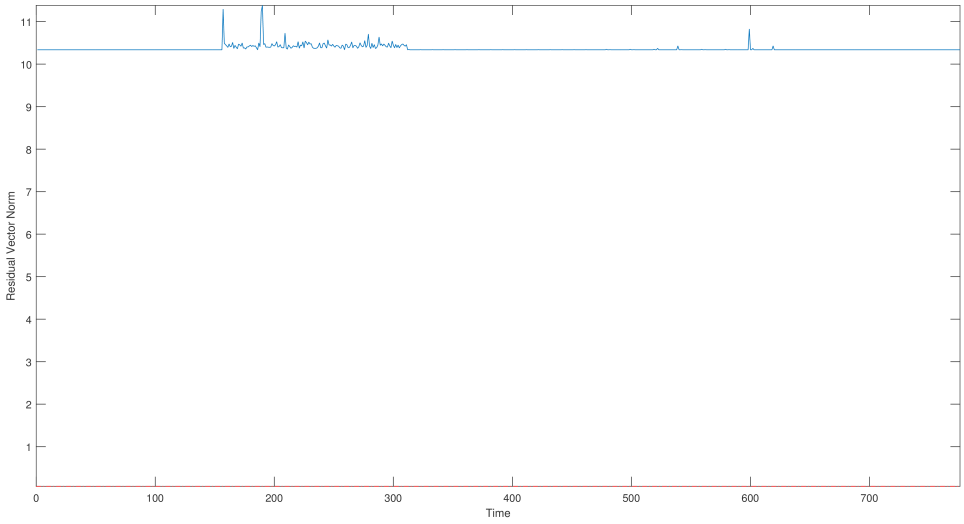


Figure 5.16: time series plots of residual vector with Q test result

PCA successfully constructs the separated subspace of the original data set and the patterns in two subspace gives strong hints about anomalies, but the Q-statistic is not appropriate as identified threshold for our test data, which is in different value range. If insist on using subspace detection thought, the setting of threshold needs to be further analyzed for identification. For example, analyze the value range in residual vector or considering value distribution from statistic scope. Or the subspace construction can be further combined with other detection techniques for identifying anomalies. Based on the result in Figure 5.15, the fluctuation of value is in the considerable range while there are several suspicious pulses. The false positive rate exists, because the true anomaly happens around index 600, which appears to a surprisingly pulse. But from index 150 to index 300, the pattern is also suspicious. The comparison from normal and residual state vector is an assistant here, which could tell if the suspicious pattern in residual vector contributes to the normal pattern or not. Figure 5.17 demonstrates this point, the green circle marked suspicious anomaly in residual vector,

which contributes to the pattern in original vector. The true anomaly is marked with red circle, which has no patterns in normal and original vector.

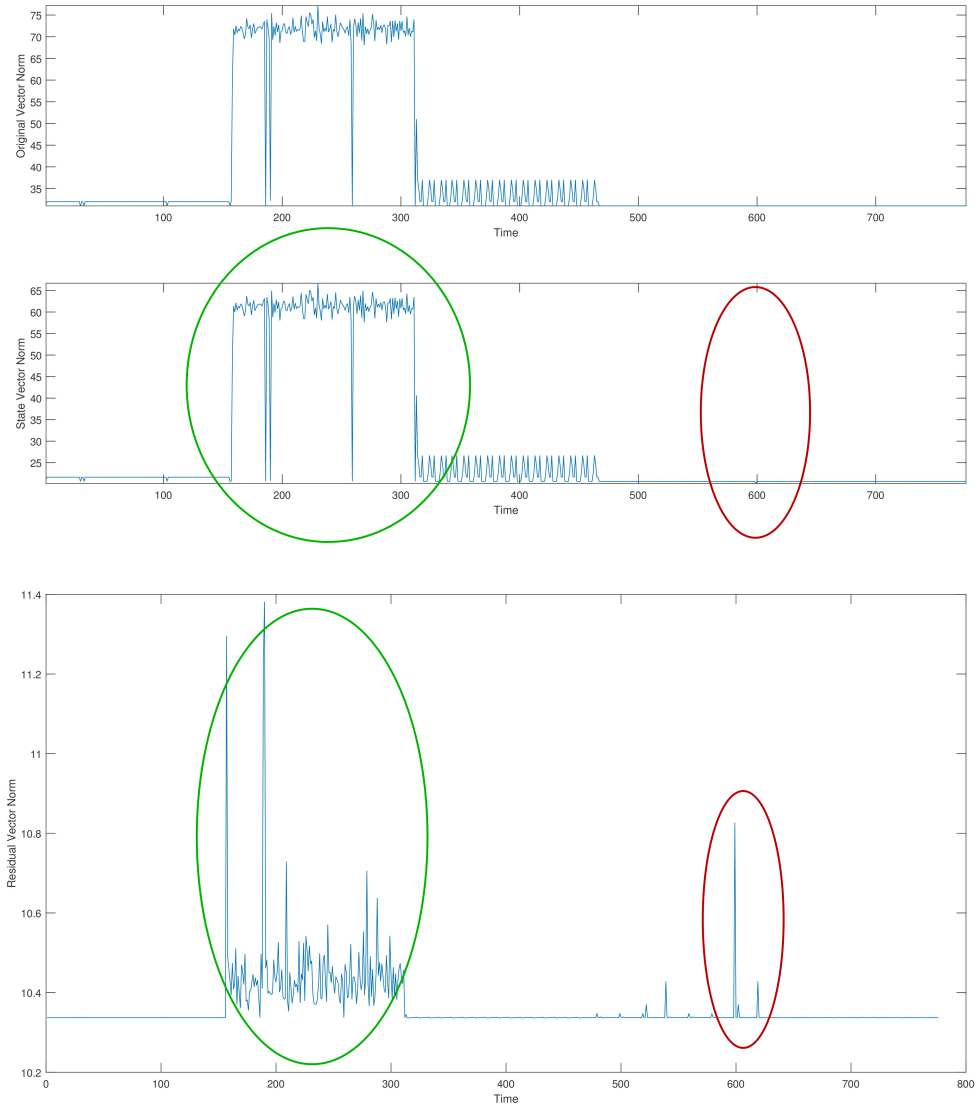


Figure 5.17: Comparison of Patterns, which indicates the true anomaly

5.3.2. K-MEANS

EXPERIMENTAL PROCESS

When implementing K-means clustering for measuring the similarity, every second in the multivariate time series is treated as a data instance. The K has been set from 3 to 20. There are two data sets, one is the normal data set, the other one is the abnormal data set, containing the error of meta data stuck. Both data sets have the same time length which lasts 775 seconds. The normal data set has been trained though K-means clustering, with K changing from 3 to 20. For each cluster, the maximum distance, also the radius of the cluster, is computed. For every data instance in the abnormal data set, its distance to each centroid is computed in order to compare with the radius. If its distance to every centroid is larger than the radius, then this time point is excluded from the normal clusters, which means it is an anomaly point.

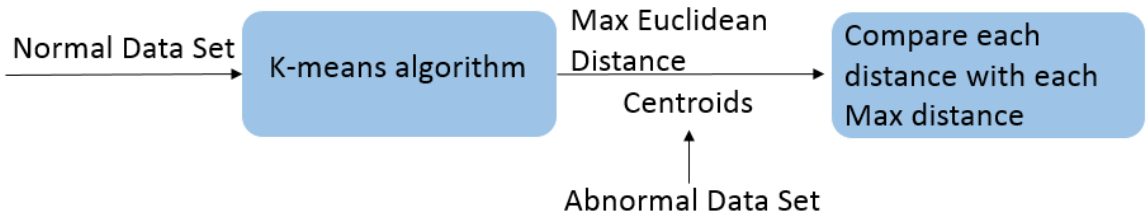


Figure 5.18: Experimental process based on K-means clustering

K-MEANS RESULT

For the input data set, K-means clustering has firstly been implemented for a general understanding about individual time unit characters. During the experiment, the location and amount of the centroid influences the result most. For the training data, the number of centroids increases from 3 to 20, and the number of unusual time units according to the Euclidean distance comparison varies, which is demonstrated in the Table 5.6. K stands for the amount of clusters and N stands for the amount of anomaly time unit index. In other words, N stands for the quantity of identified anomalies, which is changing with K changing.

K	3	4	5	6	7	8	9	10	11
N	4	4	1	5	12	5	4	4	4
K	12	13	14	15	16	17	18	19	20
N	4	20	20	20	20	20	20	20	20

Table 5.6: quantity of the clusters and anomaly units

The amount of anomaly index fluctuates with the changing amount of initial centroids, which is an unexpected feedback. When the amount of anomalies is equal, same identification result is obtained rather than random results. According to the test result, the anomaly justified in every group is index 622. Take 3 centroids for instance, the largest distance of each cluster is 4.2706, 1.7159 and 6.1535, the distance for index 622 to each centroid is 5.3018, 8.9196 and 6.2469. In other words, though it has further distance than the largest distance of each cluster, according to the data, index 622 is almost on the edge of the third cluster, with a slight difference at 10^{-1} magnitude. In order to check if the failure is rooted from improper algorithm, a simple test for normal training data set clustering has been implemented. Take 3 centroids as example, Table 5.7 below contains the largest distance of each cluster and the distance between each two centroids follows the order of cluster 0 and 1, cluster 1 and 2, cluster 0 and 2.

cluster	0	1	2
Largest dis	4.2706	1.7159	6.1535
Distance	6.6609	6.2404	2.2140

Table 5.7: description of data components

Apparently, the pairwise distance among centroids and their largest distance reflects the coverage among different clusters. If the amount of centroids reduces to 1, the largest distance of the cluster is 6.3547 and there is still one anomaly identified with the distance to the centroid is 6.3555, with a slight difference at 10^{-3} magnitude. After checking the data set, it turns out that in that second, one kind of categorical attribute lost toggling. When $K = 1$ in the normal training data set, it is possible to detect time unit anomaly, but both the sensitivity and accuracy seem to be low, which is not qualified enough for our research.

5.3.3. k-NN

EXPERIMENTAL PROCESS

Only the data set contains error of meta data stuck is applied in this section. When applying for k-NN to measure the similarity, every second has been viewed as data instance. Each data instance has the quantity

of k nearest neighbours' distance every computing time. The sum, maximum, minimum and average of this group of distances for every data instance is computed in case different parameters give different results. The patterns visualized in the figure respect the changing of group distances' over time. When the group distances' result tends to be higher in value range, the more suspicious of being anomaly the point is.

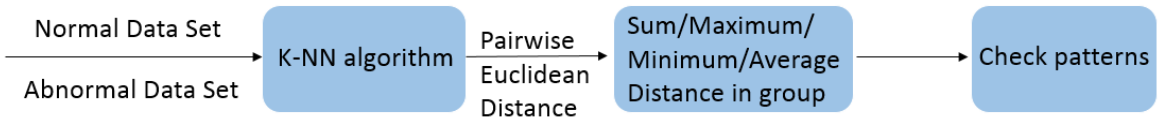


Figure 5.19: Experimental process based on k-nn

k-NN RESULT

Four kinds of nearest neighbours' distance have been calculated and based on a $k = 3$, the result in Figure 5.20 proves that all these four parameters can be applied for anomaly detection because they share the similar corresponding patterns among each other. Though minimum distance seems to have small differences with the rest three parameters, it gives more general patterns.

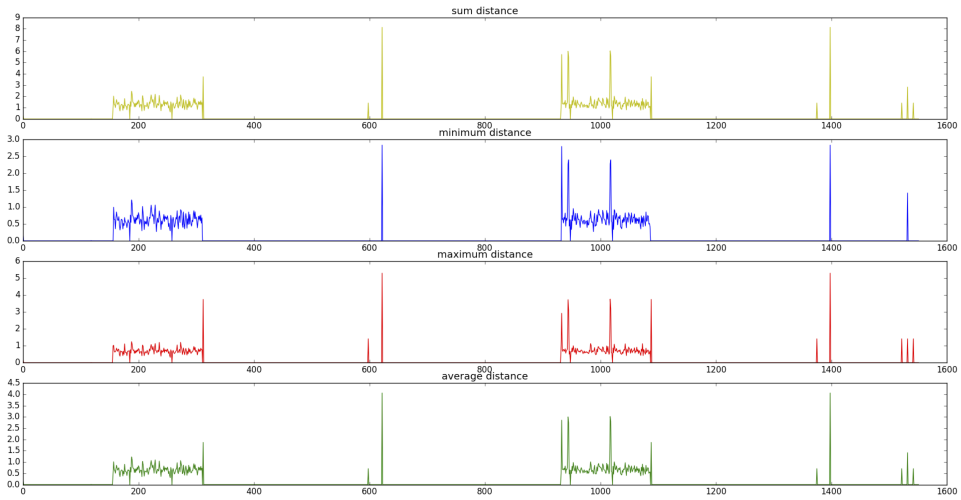
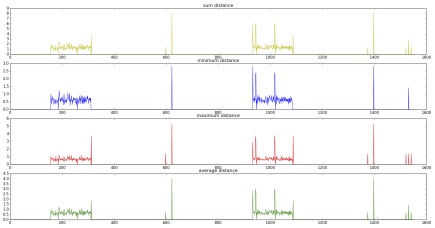
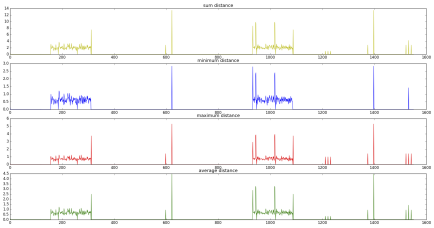


Figure 5.20: the distance measured of k-NN, with k=3

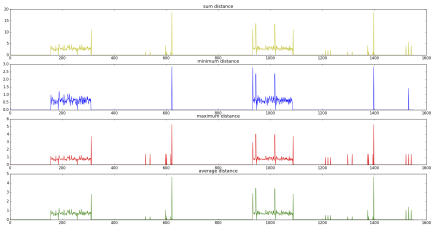
Besides the parameters in each group (nearest neighbours' distance for each data point), the scale of k is also an influential factor for detecting anomalies. With k changing from 3 to 10, the result displays by order as following Figure 5.21. With the increasing of k, the scale itself keeps stable in the range. The duration of an anomaly becomes obvious, the surprising pulse becomes more obvious. In another word, kNN can be used for anomaly detection and the bigger k is, the more characters about the anomaly can be demonstrated directly by trend of distance change. The setting of k and the threshold distance needs to be adjusted for certain system.



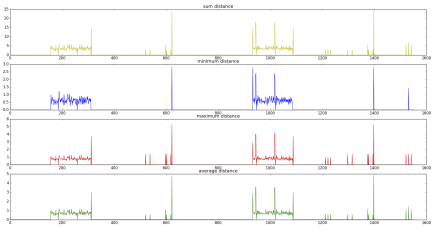
(a) k=3



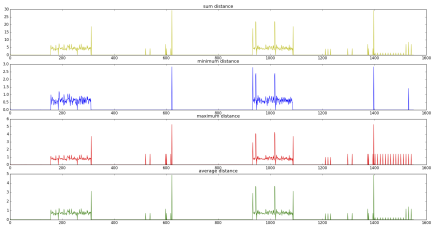
(b) k=4



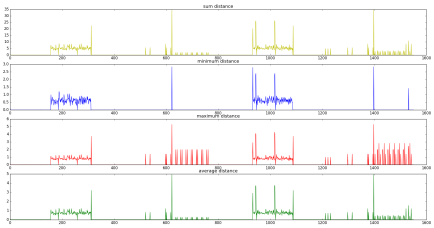
(c) k=5



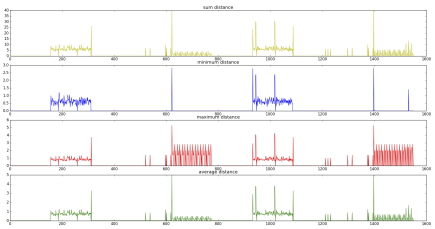
(d) k=6



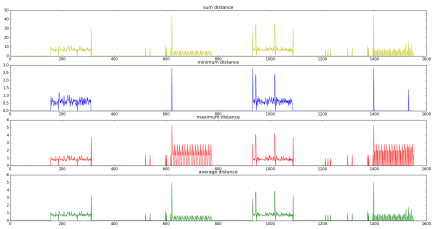
(e) k=7



(f) k=8



(g) k=9



(h) k=10

Figure 5.21: the result of kNN, k from 3 to 10

Based on the same distance measurement, k-NN displays better performance on detecting anomalies than K-means, which failed to identify the anomalies. The big difference between K-means clustering and k-NN is that the K-means clustering evaluate each instance with respect to the cluster it belongs to, while k-NN analyze each instance based on its local neighborhood. This is why k-NN gives more accurate result about similarity compared with K-means clustering. The right half part of k-NN shares the similar patterns with PCA. The suspicious impulses actually are around index 600 and 1400. Though the threshold value is not set, it is obvious that there exists false positive in the pattern around index 1000. Take k as 3 for example, the real anomaly starting point is marked in red circle, while the false positive is marked in green circle Figure 5.22.

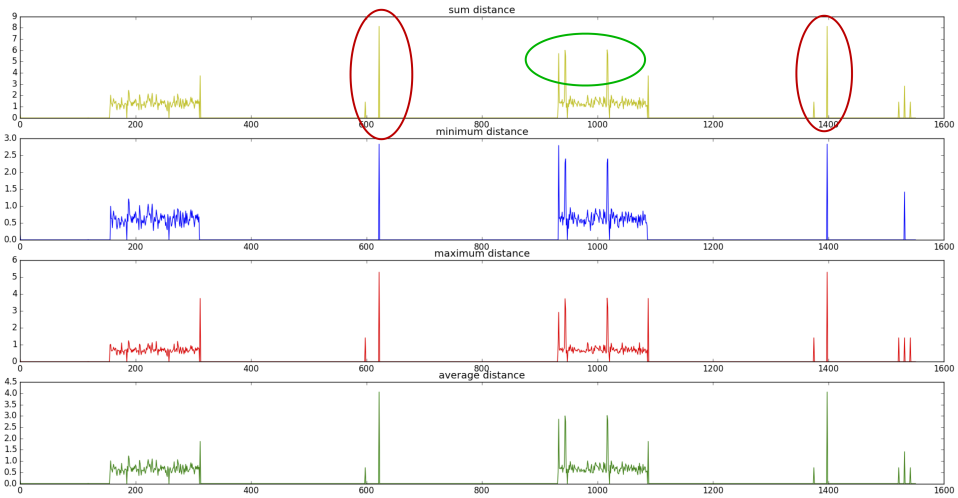


Figure 5.22: k-NN result analysis

In conclusion, if we only compare three general parameters of all the techniques applied, the result is shown as Table 5.8. The three parameters are identification, location, and classification. In other words, whether the technique can detect, locate, and classify the anomalies. These parameters are based on the test result with corresponding data sets. RTI+ has the best performance with discrete event sequences as the input. Grammarviz behaves normally with univariate time series as the input. When multivariate time series as the input, PCA and k-NN behaves similarly and the threshold needs to be further analyzed.

Technique	Identification	Location	Classification
N-gram	✓	×	–
RTI+	✓	✓	✓
Grammarviz	✓	✓	–
PCA	✓	✓	–
K-means	×	×	–
k-NN	✓	✓	–

Table 5.8: comparison among techniques based on three parameters

6

CONCLUSION

This research focuses on detecting anomalies in the DVB system. Based on varied data sets, different corresponding methods have been tested for this purpose. For discrete event sequence, RTI+ apparently gives good explanation and performs very well. RTI+ can not only detect anomalies, but also classify the anomalies type with rejection, which gives the location of happening error. For similar discrete event sequences, they can be trained together to obtain an automata which dealing with this kind of input. The only flaw about RTI+ is that it has limitation about the input format, the data has to be segmented into readable frames and the time has to be adjusted with certain precision level in integer. N-gram can also detect the existence of anomalies, but comparatively, N-gram has limitation for the anomaly types and it cannot locate the error directly. For univariate time series, grammarviz which is based on SAX and context-free grammar gives description about related grammars for both usual patterns and anomalies in the time series. But its performance is not good enough as expected. The false positive is hidden behind the algorithm, which cannot be improved with background knowledge of system. Changing the parameters of the algorithm, such as window size, alphabet size will influence the test result, but it will influence the time consuming at the same time. Also, the anomaly of information lost along a long time period seems hard to be detected. Because its capability of handling long period data is limited, though it has higher sensitivity about patterns' change. For multivariate time series, PCA explains the original pattern construction based on subspaces thoughts. The threshold in PCA can be adjusted and combined with background knowledge of the system in order to obtain better performance. It indicates another feature that the multivariate time series can be discomposed into groups for detecting anomalies, which probably could give valuable results. According to the test result, K-means clustering is not appropriate for detecting anomalies because the similarity is not separable in this way. K-NN demonstrates that the parameters of group distance (sum, maximum, minimum, average) have high consistency, and it can be adjusted for detecting anomalies with bearable false positive rate. Similarly as PCA, the threshold of k-NN could be further analyzed and combined with the background knowledge so that the result is more persuasive.

In general, this research tests varied methods for multiple data sets and provides suggestions for targeted anomaly types. Large amount of preprocessing steps have been applied for handling every case, which overcomes the challenges from the data set. For the unequal sampling time and mixed types of data, filling the empty values, one-hot coding, and normalization have been applied to handle this obstacle. If without any prior knowledge of the system, it would be better to first analyze the characters of the data set and take a small step to reduce its dimensions. Based on the difficulty of this research, a prior background knowledge is a big assitant for detecting anomalies and anlayzing patterns. The difficulty of detecting anomalies can be reduced starting from the data collection process. The data can be extracted and collected in an easier way for data analyst to provide avaliable solutions. RTI+ performs best corresponding to the discrete event sequence, which can detect and classify anomalies with low false positive. The univariate time series could be analyzed with other algorithms for better result. However, for the multivariate time series, it seems that to analyze it as a whole case is not a best choice, though PCA and k-NN performs as expected. But all of these methods are not so appropriate for predicting the happening of anomalies or finding the hidden patterns before anomalies' appearance. It would be even harder to make prediction on real-time monitoring data streaming. Based on the data sets applied in this research, it is hard for starters to detect anomalies directly without any background knowledge of the DVB system. The better understanding about the DVB system should be

more helpful for collecting the data. And it is more efficiency and accurate if the data collection and targeted anomalies are more specific, rather than a general detection technique which can cover all the possibilities. In this research, combined with data types, generative models such as RTI+ are more appropriate for low dimension data sets, which could generate and capture the features of the data set. Discriminative models have more capability of handling high-dimension data sets.

There are three directions which can be developed in the future research. The first one is to decompose the multivariate time series efficiently. It would be better if it can be collected in the way of discrete event sequence, univariate time series, and so on. In other words, to chase for the generated algorithm which can detect and even predict anomalies, it would be more efficient and accurate to start with smart data collection. Filter and collect the monitoring data with goals which is adequate preparation for further analysis. The other direction is to boost the potential detection techniques for targeted anomalies. And also the setting of the threshold should be combined with background knowledge of the system, which is an important subdirection. This is a wide direction requires adquate For example, all the discrete sequence events can be collected for RTI+ to learn a general automata for anomaly detection. The only problem is that the adaptive capability of the data mount may cause some limitations for implementation. In this case, the background knowledge of the system is not so influential, which could save time and resources for starters. But for other techniques, such as PCA in this research, the threshold setting needs further analyzed for identifying and even classifying the anomalies. The third research direction is a further step based on the first two direction, which is putting the experimental result in real-time data collection and anomaly detection. When considering dynamic data stream and real-time detection, the requirement for the algorithms becomes more complicated and also the efficiency should not be reduced.

BIBLIOGRAPHY

[1] F. Palmieri, U. Fiore, A. Castiglione, and A. De Santis, *On the detection of card-sharing traffic through wavelet analysis and support vector machines,* Applied Soft Computing **13**, 615 (2013).

[2] V. Chandola, A. Banerjee, and V. Kumar, *Anomaly detection: A survey,* ACM computing surveys (CSUR) **41**, 15 (2009).

[3] P. PhysioBank, *Physionet: components of a new research resource for complex physiologic signals,* Circulation. v101 i23. e215-e220 .

[4] G. L. Tietjen and R. H. Moore, *Some grubbs-type statistics for the detection of several outliers,* Technometrics **14**, 583 (1972).

[5] S. Verwer, M. de Weerdt, and C. Witteveen, *Efficiently identifying deterministic real-time automata from labeled data,* Machine learning **86**, 295 (2012).

[6] J. Lin, E. Keogh, L. Wei, and S. Lonardi, *Experiencing sax: a novel symbolic representation of time series,* Data Mining and knowledge discovery **15**, 107 (2007).

[7] F. Palmieri, U. Fiore, A. Castiglione, and A. De Santis, *On the detection of card-sharing traffic through wavelet analysis and support vector machines,* Applied Soft Computing **13**, 615 (2013).

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

[9] P. Sun, S. Chawla, and B. Arunasalam, *Mining for outliers in sequential databases.* in *SDM* (SIAM, 2006) pp. 94–105.

[10] C. C. Noble and D. J. Cook, *Graph-based anomaly detection,* in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (ACM, 2003) pp. 631–636.

[11] S. Shekhar, C.-T. Lu, and P. Zhang, *Detecting graph-based spatial outliers: algorithms and applications (a summary of results),* in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (ACM, 2001) pp. 371–376.

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

[13] F. E. Grubbs, *Procedures for detecting outlying observations in samples,* Technometrics **11**, 1 (1969).

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

[15] J. Zhang and H. Wang, *Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance,* Knowledge and information systems **10**, 333 (2006).

[16] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, *Lof: identifying density-based local outliers,* in *ACM sigmod record*, Vol. 29 (ACM, 2000) pp. 93–104.

[17] A. K. Jain and R. C. Dubes, *Algorithms for clustering data* (Prentice-Hall, Inc., 1988).

[18] D. Barbará, Y. Li, J. Couto, J.-L. Lin, and S. Jajodia, *Bootstrapping a data mining intrusion detection system,* in *Proceedings of the 2003 ACM symposium on Applied computing* (ACM, 2003) pp. 421–425.

[19] I. Jolliffe, *Principal component analysis* (Wiley Online Library, 2002).

[20] H. Dutta, C. Giannella, K. D. Borne, and H. Kargupta, *Distributed top-k outlier detection from astronomy catalogs using the demac system.* in *SDM* (SIAM, 2007) pp. 473–478.

[21] A. Bhattacharya, A. G. Parlos, and A. F. Atiya, *Prediction of mpeg-coded video source traffic using recurrent neural networks,* IEEE Transactions on Signal Processing **51**, 2177 (2003).

[22] M. Ellis, D. P. Pezaros, T. Kypraios, and C. Perkins, *A two-level markov model for packet loss in udp/ip-based real-time video applications targeting residential users,* Computer Networks **70**, 384 (2014).

[23] J. Shin and P. C. Cosman, *Classification of mpeg-2 transport stream packet loss visibility,* in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing* (IEEE, 2010) pp. 910–913.

[24] S. Verwer, M. de Weerdt, and C. Witteveen, *A likelihood-ratio test for identifying probabilistic deterministic real-time automata from positive data,* in *International Colloquium on Grammatical Inference* (Springer, 2010) pp. 203–216.

[25] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung, *Similarity search for multidimensional data sequences,* in *Data Engineering, 2000. Proceedings. 16th International Conference on* (IEEE, 2000) pp. 599–608.

[26] Q. Lin, J. Wang, and W. Qiao, *Denoising of wind speed data by wavelet thresholding,* in *Chinese Automation Congress (CAC), 2013* (IEEE, 2013) pp. 518–521.

[27] Q. Lin, C. Hammerschmidt, G. Pellegrino, and S. Verwer, *Short-term time series forecasting with regression automata,* in *ACM SIGKDD 2016 Workshop on Mining and Learning from Time Series (MiLeTS)* (2016).

[28] D. Cheboli, *Anomaly detection of time series*, Ph.D. thesis, University of Minnesota (2010).

[29] D. Harris and S. Harris, *Digital design and computer architecture* (Elsevier, 2012).

[30] Y. Sakakibara, *Recent advances of grammatical inference,* Theoretical Computer Science **185**, 15 (1997).

[31] A. Stevenson and J. R. Cordy, *A survey of grammatical inference in software engineering,* Science of Computer Programming **96**, 444 (2014).

[32] N. Chomsky, *Three models for the description of language,* IRE Transactions on information theory **2**, 113 (1956).

[33] W.-J. Li, K. Wang, S. J. Stolfo, and B. Herzog, *Fileprints: Identifying file types by n-gram analysis,* in *Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop* (IEEE, 2005) pp. 64–71.

[34] K. J. Lang, B. A. Pearlmutter, and R. A. Price, *Results of the abbadingo one dfa learning competition and a new evidence-driven state merging algorithm,* in *International Colloquium on Grammatical Inference* (Springer, 1998) pp. 1–12.

[35] K. J. Lang, B. A. Pearlmutter, and R. A. Price, *Results of the abbadingo one dfa learning competition and a new evidence-driven state merging algorithm,* in *International Colloquium on Grammatical Inference* (Springer, 1998) pp. 1–12.

[36] Y. Guédon, *Estimating hidden semi-markov chains from discrete sequences,* Journal of Computational and Graphical Statistics **12**, 604 (2003).

[37] W. L. Hays, *Statistics, holt,* Rinehart and Winston, New York , 343 (1988).

[38] S. S. Wilks, *The large-sample distribution of the likelihood ratio for testing composite hypotheses,* The Annals of Mathematical Statistics **9**, 60 (1938).

[39] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, *Fast subsequence matching in time-series databases*, Vol. 23 (ACM, 1994).

[40] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, *Locally adaptive dimensionality reduction for indexing large time series databases,* ACM SIGMOD Record **30**, 151 (2001).

[41] K.-P. Chan and A. W.-C. Fu, *Efficient time series matching by wavelets,* in *Data Engineering, 1999. Proceedings., 15th International Conference on* (IEEE, 1999) pp. 126–133.

[42] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, *Locally adaptive dimensionality reduction for indexing large time series databases,* ACM SIGMOD Record **30**, 151 (2001).

[43] https://www.cs.rochester.edu/~nelson/courses/csc_173/grammars/cfg.html, .

[44] C. G. Nevill-Manning and I. H. Witten, *Identifying hierarchical strcture in sequences: A linear-time algorithm,* J. Artif. Intell. Res.(JAIR) **7**, 67 (1997).

[45] P. Senin, J. Lin, X. Wang, T. Oates, S. Gandhi, A. P. Boedihardjo, C. Chen, S. Frankenstein, and M. Lerner, *Grammarviz 2.0: a tool for grammar-based pattern discovery in time series,* in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (Springer, 2014) pp. 468–472.

[46] H. Abdi and L. J. Williams, *Principal component analysis,* Wiley Interdisciplinary Reviews: Computational Statistics **2**, 433 (2010).

[47] A. Lakhina, M. Crovella, and C. Diot, *Diagnosing network-wide traffic anomalies,* in *ACM SIGCOMM Computer Communication Review*, Vol. 34 (ACM, 2004) pp. 219–230.

[48] J. E. Jackson and G. S. Mudholkar, *Control procedures for residuals associated with principal component analysis,* Technometrics **21**, 341 (1979).

[49] R. Dunia and S. J. Qin, *Multi-dimensional fault diagnosis using a subspace approach,* in *American Control Conference* (1997).

[50] R. Dunia and S. Joe Qin, *Subspace approach to multidimensional fault identification and reconstruction,* AIChE Journal **44**, 1813 (1998).

[51] C. A. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, M. Vlachos, and G. Das, *Mining time series data,* in *Data mining and knowledge discovery handbook* (Springer, 2009) pp. 1049–1077.

[52] T. W. Liao, *Clustering of time series data?a survey,* Pattern recognition **38**, 1857 (2005).

[53] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, *Data stream mining,* in *Data Mining and Knowledge Discovery Handbook* (Springer, 2009) pp. 759–787.

[54] P. Berkhin, *Survey of clustering data mining techniques: Technical report. accure software,* (2003).

[55] J. M. Pena, J. A. Lozano, and P. Larranaga, *An empirical comparison of four initialization methods for the k-means algorithm,* Pattern recognition letters **20**, 1027 (1999).

[56] D. J. MacKay, *Information theory, inference and learning algorithms* (Cambridge university press, 2003).

[57] F. Aurenhammer, *Voronoi diagrams-a survey of a fundamental geometric data structure,* ACM Computing Surveys (CSUR) **23**, 345 (1991).

[58] N. S. Altman, *An introduction to kernel and nearest-neighbor nonparametric regression,* The American Statistician **46**, 175 (1992).

[59] P.-N. Tan *et al.*, *Introduction to data mining* (Pearson Education India, 2006).

[60] D. R. Jensen and H. Solomon, *A gaussian approximation to the distribution of a definite quadratic form,* Journal of the American Statistical Association **67**, 898 (1972).