Real-time Intrusion Detection of Cyber Physical Systems

Version of September 28, 2020

Kaixin Ding

Real-time Intrusion Detection of Cyber Physical Systems

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Kaixin Ding
born in Shanghai, People's Republic of China

Cyber Security Group
Intelligent System
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl

Real-time Intrusion Detection of Cyber Physical Systems

Author: Kaixin Ding
Student id: 4623649
Email: dingkx12@gmail.com

Abstract

Intrusion detection problem in Industrial Control Systems(ICS), such as water treatment plant and power grid, is an important real-world problem. Real-time anomaly detection have been proposed to minimize the risk of cyber attack. In this study, two different kind of intrusion detection mode-based approach are learned from normal behaviour of an ICS, SWaT. One model is inspired by TABOR, a offline graphical model-based approach of CPS intrusion detection. Timed automaton, Bayesian network and Out of Alphabet are combined to find anomalies and localize the abnormal sensors and actuators. The other model proposed is based on two time slice Bayesian Network(2-TBN) with the motivation simplifying multi-model approach into a single model one. In this way, cost of computing power and time is reduced in order to meet the requirement of real-time operation testing.

Experimental results demonstrate the model's close performance to TABOR and slight advantage on time. The study about difference between offline testing and real-time operation is another topic in this study. The generic underlying idea and experience is also applicable to cyber physical system in other industrial control systems.

Keywords: Intrusion detection, Industrial Control System, Cyber Physical System, Timed automaton, Bayesian Network, Two time slice Bayesian Network, Real-time operation testing.

Thesis Committee:

Chair:	Prof.dr.ir. R.L. (Inald) Lagendijk, Faculty EEMCS, TU Delft
University supervisor:	Dr.ir. S.E.(Sicco) Verwer, Faculty EEMCS, TU Delft
Committee Member:	Dr.ir. Huijuan Wang, Faculty EEMCS, TU Delft

Preface

After a quite long-term efforts, with great happiness, I finally present this work as the fulfillment of my master's study.

This work cannot be accomplished without the help of many people. Thanks to my supervisor Sicco Verwer, for introducing me into this interesting study topic and offering precious guidance in our weekly meeting. With his help, I know the right way of doing research and have the courage to restart when I was stuck in study.

Thanks to PhD Qin Lin, my co-supervisor, for a lot of help in both study and life. He sets a good example to me and keeps giving me the fruitful discussions and suggestions even when he is busy with his own work. Thanks all the friends who are also supervised by Sicco. Our weekly group discussion really provides me with many valuable ideas in experiment design and thesis writing.

Thanks to my friend Shijian Zhong, Yikai Lan and Xiangwei Shi, for the dinner, beer, and talk we have together.

Thank my parents and all my family back in China for their support and encouragement throughout my master study.

Finally, I would like to thank myself. I went through several big changes in life during this master thesis. My respectable grandfather passed away. Girlfriend of four years betrayed me and left. My mother keeps fighting with severe bone disease. All these things makes me become sensitive and hard to concentrate. However, with the help of all the people I mentioned above, I finally fulfill my master study. Now I can say undoubtedly that it is a journey with abundant harvest!

Kaixin Ding
Delft, the Netherlands
September 28, 2020

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Industrial control system (ICS) is a general term that encompasses several types of control systems and associated instrumentation used for industrial process control [38]. Nowadays ICSs are widely used for public infrastructure, such as power generation, water treatment, gas processing and telecommunications, which is critical to any society. Due to great damage a potential attack on an ICS could cause, researchers pay more and more attention on the protection of ICSs these days.

ICSs systems are usually implemented by the Supervisory Control and Data Acquisition (SCADA) workstations, Human Machine Interface (HMI), programmable logic controllers (PLCs) and communication network which underlies the systems. In most cases, A system is vulnerable to attacks due to the presence of all cyber components mentioned above[45]. BY using one or more known vulnerabilities in an ICS, attackers can easily launch an attack and cause significant damage.

Such vulnerabilities can be reflected in variable forms:

- Lack of access control: Lack of access control in an ICS can lead to the compromise of ICS's security [3].
- Software vulnerabilities: SCADA and PLCs in ICSs rely on software system to a large extent. As a result, ICSs may also suffer from vulnerabilities caused by software.
- Network vulnerabilities: ICSs often includes an underlying network which is used for internal and external communication. Therefore vulnerabilities of communication network may be also used to launch attacks.

Attacks on ICSs can cause various types of losses, even cascading failures [24], which can harm large communities, such as cities and countries. An attack targeted at an ICS for power generation can cause a widespread blackout which could cause countless monetary losses.

Attacks on ICSs, especially ones which targets critical infrastructure, are increasing each year. Both successful attacks[9, 27, 44] and unsuccessful attempts recorded point out the utmost importance of research in security of ICS.

Intrusions detection is an effective way to help people record and react rapidly to the attacks. The result of detection gives out the instruction of the research in security of ICS. Therefore there is an urgent need to design a real-time intrusions detection system for the goal of making ICS invulnerable to cyber attacks.

As ICS includes both cyber physical systems(CPS), such as PLCs, and underlying communication network, the intrusions detection problem can be also considered as the combination of two parts. In this research, most efforts are focused on detecting intrusions in the physical process of ICS due to limited time and resources. Network security problem about underlying communication network hasn't been taken into account yet. In other words, the emphasis of this research is laid on real-time cyber attacks detection in only CPS of ICS.

1.2 Intrusions detection problem in CPS

Among increasing researches about cyber attacks detection, there are four main methods arousing most interest and widely used in different research and operation cases. They are signature-based detection[15, 34], verification[8, 47], behavior specification-based detection[2], and machine learning[16, 23]. These four methods varies in many aspects and have their own strengths and weaknesses.

1.2.1 Signature-based detection

The core of signature-based method is to obtain an up-to-date signature dictionary of all known attacks. The dictionary can be retrieved at any time point during cyber attacks detection. It is one of the most traditional method in cyber attacks detection. Any known attack recorded on the dictionary can be easily pointed out and the detection procedure costs little when the dictionary is in finite size.

However, nowadays attackers also keep working on developing new techniques which cause growing number of unknown threats. Due to this fact, signature-based detection is becoming increasingly infeasible and unreliable. People start turning to more advanced methods other than this traditional dictionary-based one.

1.2.2 Verification

In verification methods, formal models are set up according to the system's design. The cost of this procedure can be very expensive due to the complexity of systems. After setting up the model, the test will be carried out on a source code level. And the detection system can give out alarms when certain signals show large deviations from the values specified in

the system's design.

The weakness of verification is apparent just as its advantage. Although verification methods can be very powerful due to his source code-based design, it can also be very difficult to analyze when resulting model is too large. With the development of industrial control systems, these state-explosion problem becomes more and more common because of the increasing system complexity.

1.2.3 Behavior specification-based detection

Behavior specification-based detection is built on the base of expert knowledge. A wide knowledge required here includes a precise understanding of how cyber physical systems behave. The procedure of obtaining this knowledge can be regarded as the most expensive procedure among all four methods mentioned. However, it can detect a lot of different attacks because the detailed models used in this method describe the underlying physical process. The detection based on expert knowledge can show the essence of cyber physical systems.

Besides expensive cost, another apparent weakness of behavior specification-based detection is its high sensitiveness to noise. As dynamic operating environment can hardly considered in modeling based on expert knowledge, modeling errors can be caused by many different situation, such as aging or other evolvements of the facility in question.

1.2.4 Machine learning

Machine learning approach is the most popular method among these four detection methods. The basic idea of machine learning is to set up models which can describe cyber physical systems in feature space. The modeling procedure of machine learning cost much less compared with the procedure of verification or behavior specification-based detection. When detecting data points with large deviation from the normal feature space in model, the machine learning-based detection system can give out alarms. In this way, a large range of attacks can be detected with low cost.

However, due to the increasing complexity of algorithms and the development of deep learning, one shortcoming of machine learning method becomes more and more apparent. As the model is set up on feature space, the detection system provides little insight into the system. No explanation of detection can be given out along with alarms in intrusion detection.

To respond to this weakness, Lin[26] proposed a graphical model-based approach for anomaly detection in industrial control systems. And the basic idea of our research in this thesis is inspired by his paper.

1.3 Problem statement

According to Lin's paper, his research aims to respond two key challenges in applying machine learning approach to solve intrusion detection problems in cyber physical systems. These two challenges are the explanation of outcome and localization of anomaly. He thinks these two questions are quite important in real-world detection problems of industrial control system because operators need these information to undertake follow-up actions after getting the anomaly alarm. In his opinion, detection is only the first step against attacks, and the explanation and localization are the same important in these real-world problems.

To deal with these problems, Lin proposed an insightful graphical model, which is named as TABOR in his paper[26]. Figure1.1 shows the flowchart of method used in TABOR.

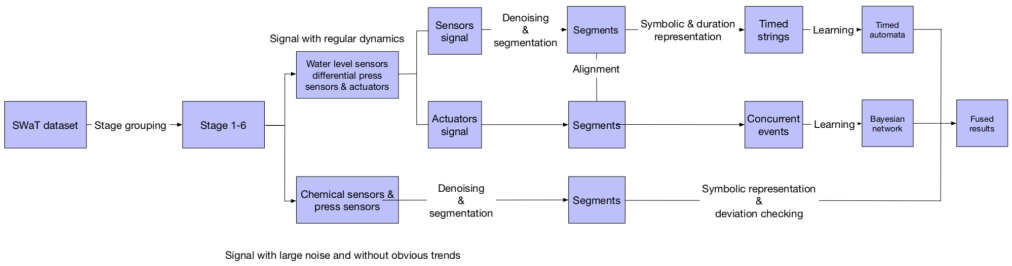

Figure 1.1: Flowchart of TABOR, figure by Lin[26]

TABOR model is learned from the normal operational observation of a scaledown water treatment system. Lin divide sensors and actuators of this system into different groups according to their different functions. The machine learning strategy he uses is a combination of timed automata(TA), Bayesian network(BN) and out-of-alphabet(OOA). TA is used to learning signals from the sensors. The trained TA model shows the underlying dynamical behavior pattern of water level and some other indexes. BN model is trained from both sensors and actuators signals. It shows the pattern of concurrent events happens in this system, which can also be considered as the dependency among events. Together with OOA, the combined model of these three techniques will detect irregular patterns and dependencies which do not match normal behavior of the system. These detected data points are considered as anomalies in this case.

According to Lin's research, the proposed TABOR model have a better general performance compared with methods based on deep neural network and support vector machine. Also, this model provides a solution for the interpretation and localization of anomalies. As this model is a graphical model, the detected anomaly can be quickly and precisely located to any sensor, actuator or abnormal process.

With deep understanding of Lin's research, some ideas are inspired and arouse my research interest. Based on Lin's research, we can find some questions and further work to do.

In Lin's research, the training and testing procedures are carried out offline. This is the first problem we find. TA is a kind of state-based method and its performance depends on accurate state division. In his offline testing, Lin generates a global perspective from the whole dataset and get a stable result of data representation. However, in real-world anomaly problems, real-time operation is very necessary to fulfill the requirement of detection. And in real-time operation, we can not get the same global perspective as Lin did. This surly increases the difficulty of maintaining system's good performance in real-time testing. So this combined machine learning strategy needs to be confirmed whether it still works in real-time operation. On the other hand, there must be differences between offline and real-time detection problems. We want to study deeply about these differences and decide what action and fix we should take in data preprocessing and modeling to increase the robustness of the whole detection system in real-time operation.

Besides the feasibility and robustness of combined strategy in real-time operation, another problem which worth concern is the reaction and testing speed of real-time operation. The reaction speed means system's frequency of alarms. In Lin's research, he drop the whole dataset into his model and get labels of all the testing data at one time. However, in real-time testing, data comes into detection system continuously and the system is required to give out labels, or alarms to every data point in time, which means the reaction and testing speed is much more important in real-time operation compared with offline version. With this requirement, the data preprocessing and modeling meets new challenges and needs new fix or technique to realize the quick and correct respond from detection system. So in our study we want to do more work at this aspect.

Based on these ideas and questions, we set up our research in this thesis.

1.4 Research Questions

The primary purpose of this research is to further study Lin's model and explore the possibility of proposing a real-time model-based detection approach which gives out quick and correct respond to real-time data. We have the following main research questions:

Can we solve real-time Intrusion Detection problem of Cyber Physical Systems with graphical model-based approach?

To structure this research, we have come up with three research questions. Each research question represents part of the research and is answered by the result of the research.

RQ1: What action should we take in data preprocessing to help increase the robustness and respond speed of detection model in real-time operation?

The first aspect we want to study is data preprocessing procedure. As we mentioned above, detection system meets new challenges in real-time operation. We want to study the differ-

ence between offline version and real-time requirement and decide what action we should take in data preprocessing to enable the proposed system to respond to incoming data more quickly and stably.

RQ2: Can we solve the real-time intrusion problem with the combined strategy of TA, BN and OOA? What is the difference between offline TABOR model and this real-time model? How does it perform?

In Lin's research, he propose a combined strategy of TA, BN and OOA as a method of solving intrusion problem with graphical model-based approach. However, as we mentioned above, all his training and testing process in offline environment. The performance of his model can not be guaranteed in real-time operation. So in modeling, we first want to implement a model for real-time intrusion detection on the base of TABOR. This modeling procedure surly have some differences with offline TABOR model due to different requirement in real-time operation. We will do necessary and reasonable changes in modeling and evaluate the final model we propose and compare it with TABOR and other machine learning method.

RQ3: Can we find any method to simplify the model and increase testing speed compared with combined strategy? Can the model newly proposed meet the requirements of both correct rate and testing speed? How does it perform?

Besides correct rate and reaction speed, any topic we focus on is the testing speed of model we propose in real-time operation. Although the multi-model approach proposed by Lin can include both concurrent and transition errors with different model which makes the detection result more comprehensive, it also arouses several times of training and testing time cost due to multiple models it uses in modeling. We want to study about the testing of TABOR and our proposed real-time model and find some method to reduce the time cost of testing. Besides confirming our newly proposed approach have an advantage in time cost over TABOR and origin model, we will also evaluate its performance of detection to make sure that this advantage is not at a great price of performance.

1.5 Outline

The outline of this paper is as follow:

- **Chapter 2: Related Work**
  Chapter 2 includes a review of related works on machine learning and techniques for detecting process anomalies in CPS.
- **Chapter 3: Data Processing**
  In Chapter 3, SWaT system and dataset are first introduced. Then the procedure of

data preprocessing is presented, which includes denoising, segmentation and alignment.

- **Chapter 4: Real-time System**
  Chapter 4 includes the construction of real-time detection system based on offline strategy and the performance comparison between offline and real-time version.

- **Chapter 5: 2-TBN System**
  Chapter 5 presents our approach which uses single technique, 2-TBN, to replace the combined strategy.

- **Chapter 6: Discussions**
  In Chapter 6, we conclude this thesis and introduce future research perspective.

Chapter 2

Related Work

In this chapter, we present a literature study on topics related to our work. As this thesis focus on presents an approach for solving real-time intrusion detection problem in cyber physical system, our literature study will focus on intrusion detection and relevant study in this topic.

This literature study includes four sections. In the first section, we introduced the definition of anomaly. Section 2 is a review of different kinds of anomaly detection techniques. As a main application of anomaly detection, we introduce the challenges of intrusion detection in section 3. Section 4 is a review of current research on intrusion detection problems of cyber physical system.

2.1 Anomaly Definition

Anomalies are patterns in data that do not conform to a well defined notion of normal behaviour[7]. Consider both its character and practical meaning, anomalies can be classified into following three categories:

- *Point Anomalies* If an individual data instance can be considered as anomalous with respect to the rest of the data, then it is termed as a point anomaly. Among all kinds of anomalies, this kind of anomaly is the simplest type of anomaly and is the focus of majority of research on anomaly detection.

- *Contextual Anomalies* If a data instance is anomalous in a specific context but is not anomalous in other conditions, then it is termed as a contextual anomaly[37]. This kind of anomalies is defined from the context environment of data which consists of contextual and behavioural attributes. In time series[43] and spatial[25] data, contextual anomalies can be commonly found. have been most commonly found in time series data. Figure2.1 shows an example of contextual anomaly. As we can see, the same temperature occur at both time t_1 and t_2. However, according to the context environment, t_2 is an anomaly while t_1 does not.

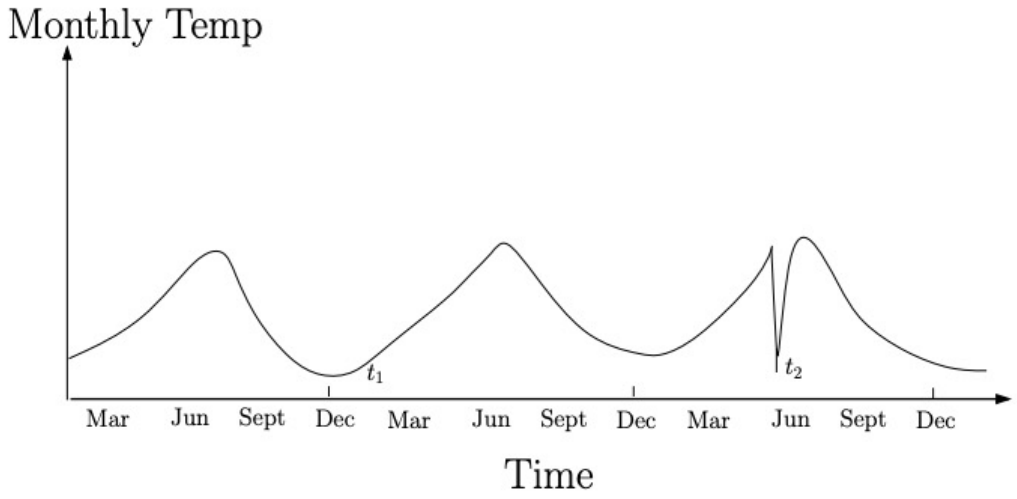

Figure 2.1: Contextual Anomaly t_2 in a Temperature Time-series, figure by Chandola[7]

- *Collective Anomalies* If a collection of related data instances is anomalous with respect to the entire data set, it is termed as a collective anomaly. Each individual data instance itself may be normal when compared to the entire dataset, but the collection of them or consistency of appearance is an anomalous situation. This kind of anomalies are commonly studied in sequence data[39], graph data[33], and spatial data[36]. Figure2.2 shows an example of collective anomaly[17]. The figure shows a human electrocardiogram output. The red region denotes an anomaly not because the value but because the abnormally long time.

2.2 Anomaly Detection Techniques

An anomaly is defined as a pattern that does not conform to expected normal behavior. And anomaly detection refers to the problem of finding these kind of patterns. With varied anomaly types, corresponding reasonable techniques for anomaly detection are also varied.

Based on the survey of anomaly detection[7], anomaly detection techniques can be categorized into the following groups:

- **Classification-Based techniques**
  Classification[13] is used to learn a model from a set of labeled data instances and classify a test instance into one of the classes using the learnt model. Common supervised learning techniques are involved, such as Neural Networks. Based on the labels available, classification can be grouped into two categories, multi-class[5] and one-class[35] anomaly detection techniques. Figure2.3 shows an example of two kinds of classification. The quality of classification depends on the accuracy and adequacy of labels.

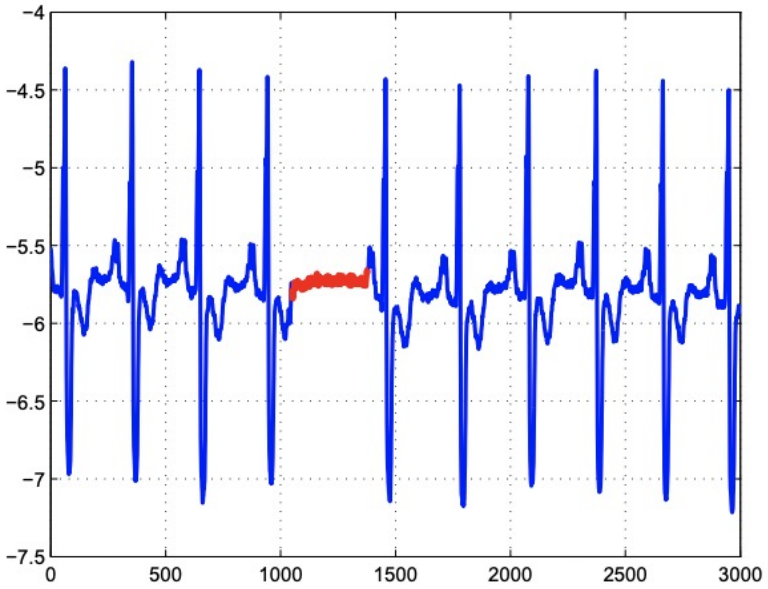

Figure 2.2: Collective anomaly corresponding to an Atrial Premature Contraction in an human electrocardiogram output, figure by Goldberger[17]

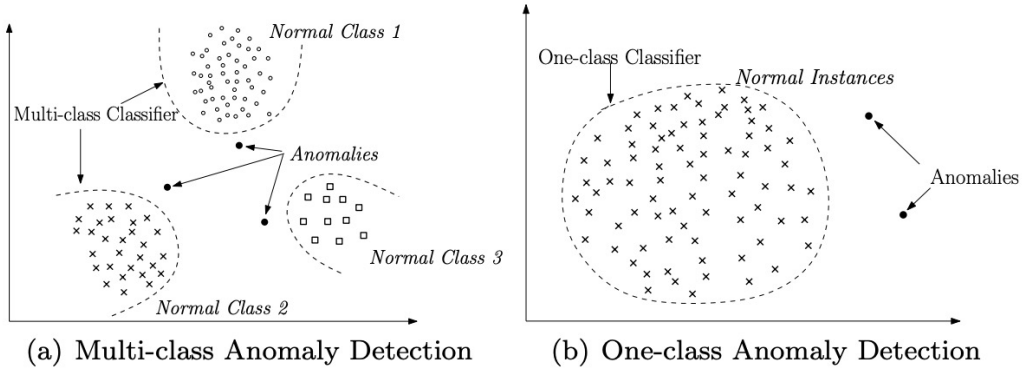

Figure 2.3: Using Classification for Anomaly Detection, figure by Chandola[7]

- **Nearest-Neighbour-Based techniques**
  The basic idea of nearest neighbor based anomaly detection techniques is that anomalies are far from their closest neighbours while normal instances occur in dense neighbours. This kind of techniques require a distance or similarity measure defined between two data instances. The distance can be computed in different methods, among which euclidean distance is one of the most common methods. Figure2.4 shows an example of Nearest neighbor based anomaly detection techniques. Both p_1 and p_2 are anomalies, but p_2 may not be distinguished from C2.

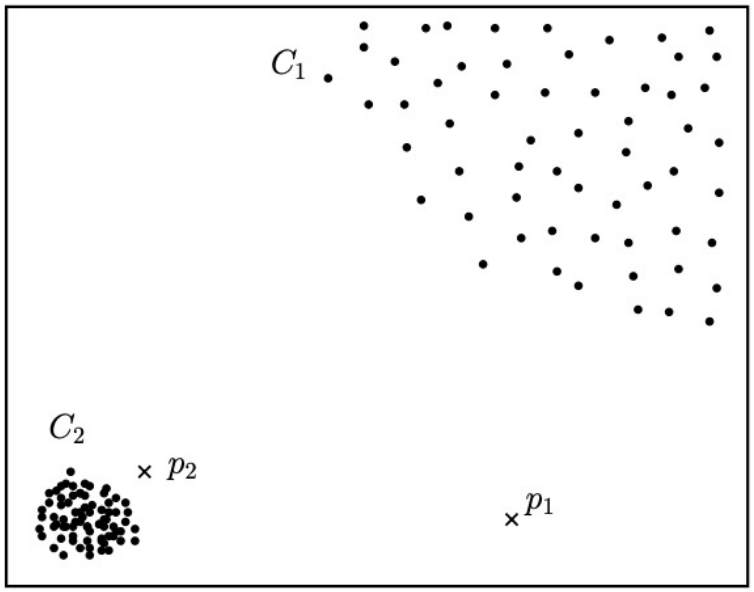

Figure 2.4: An Example of Nearest Neighbour-based Techniques, figure by Chandola[7]

It is worth mentioning that with different values of nearest neighbours, the detection result varies. In some implementation, anomalous score of instance will be defined to help improving the result[46]. The significant advantage of nearest neighbour based techniques is that this kind of techniques is naturally unsupervised and totally data driven, which have no assumption or requirement of the data structure.

- **Clustering-Based techniques**
  Clustering is to group similar data instances into clusters, which is primarily an unsupervised or semi-supervised techniques. Although clustering and anomaly detection appear to be fundamentally different, the basic concept of clustering satisfies the requirement of similarity measurement issue. Therefore, several clustering based anomaly detection techniques have been developed. These techniques are divided into three main types according to variable assumptions on data:
  - Normal data instances belong to a cluster in the data, while anomalies either do not belong to any cluster.
  - Normal data instances lie close to their closest cluster centroid, while anomalies are far away from their closest cluster centroid.
  - Normal data instances belong to large and dense clusters, while anomalies either belong to small or sparse clusters.

  Clustering-based techniques can be adapted to complex data types. The testing phase for clustering-based techniques is fast since every test instance only needs to be compared with the centroids of clusters rather than all the data. However, the efficiency of

clustering-based techniques is highly dependent on the structure of normal instances.

- **Statistical techniques**
  Statistical techniques are based on probability of data instances. It is assumed that normal data instances occur in high probability regions, while anomalies occur in the low probability regions.

  Statistical techniques learn a statistical model of normal behavior and then apply a statistical test to determine whether an instance in testing data belongs to this model. Most of these techniques are targeting at single dimension[19].

  Statistical techniques can be classified into two groups considering whether it is parametric. Parametric techniques assume the knowledge of underlying distribution and estimate the parameters from the given data[14], while non-parametric techniques do not generally assume knowledge of underlying distribution[12].

  If the assumptions regarding the underlying data distribution hold true, statistical techniques provide a statistically justifiable solution for anomaly detection. However, it is worth pointing out that a prior knowledge or reasonable guess about the data distribution is not always available in practice. On the other hand, statistical techniques are not good at dealing with dynamic behaviours with timely features.

- **Information theoretic techniques**
  In formation theoretic techniques analyze the information content of data. There are variable information theoretic meansurs, such as *Kolomogorov Complexity* and *entropy*. It is assumed that anomalies induce irregularities in the information content of the whole data.

  This kind of techniques are often used in data with naturally ordered, such as time sequential data and spatial data. And these techniques can be also operated on an unsupervised learning. However, the performance of such techniques is highly dependent on the choice of the information theoretic measure. And also, a quantity measure of anomaly, such as an anomaly score, is difficult to be associated when using an information theoretic technique.

- **Spectral techniques**
  Spectral techniques try to find an approximation of the data using a combination of attributes which capture the main variability in the original data. In such techniques, data is represented into a lower dimensional subspace in which normal data and anomalies shows great differences. Principle Component Analysis(PCA) is a

typical spectral techniques[21] for representing data into lower dimensional space.

The key advantage of spectral techniques is that it automatically perform dimension reduction and are suitable for dealing with high dimensional data. However, spectral techniques are only useful when the normal and anomalous instances are separable in the lower dimensional subspace of original data and the computational complexity is quite high.

There are other methods for classification of anomaly detection techniques. For example, techniques can be separated into supervised techniques and unsupervised techniques according to training data labels. In fact, most of the existing anomaly detection techniques solve a specific formulation of the problem, so the choice of suitable techniques is essential in anomaly detection problem.

2.3 Challenges of Intrusion Detection Problem in CPS

At an abstract level, an anomaly detection approach is to define a region representing normal behavior in the dataset and declare any observation not belonging to this normal region as an anomaly.

However, in practice several factors make this approach very challenge:

- It is very difficult to define a normal region which encompasses every possible normal behavior. In many cases, the boundary between normal and anomalous behavior is often not precise. Thus an observation close to the boundary can be misjudged.
- When anomalies are launched on purpose, attackers often make the anomaly apper like normal. In these cases, anomaly detection will be more difficult.
- It is difficult to keep the update of normal behaviour region. An out-of-date notion might not be sufficiently representative.
- Suitable dataset for training and testing is not always available.
- The noise contained in data tends to be similar to anomalies. This fact increase the difficulty to distinguish and remove noise.

Due to the above challenges, the anomaly detection problem is not easy to solve. Intrusion detection is a typical application of anomaly detection. Thus we need to face the same challenges in intrusion detection.

There are also some other specific challenges in CPS intrusion detection problem:

- Besides obtaining noise, the dataset in CPS often has high dimension and complexity. This is because CPS often includes a large number of sensors and actuators with variable functions.
- In CPS intrusion detection problem, the proposed model should provide a solution for the interpretation and localization of anomalies. In real world problem, model used in this kind of problem is often required to be visualizable and interpretable, thus enabling a better understanding of the system and verification of the model itself.

2.4 Current research

There exist several techniques for detecting anomalies in CPS. In our study, we focus on anomalies due to sensor and actuator manipulation, so we only discuss the research about this kind of anomalies.

The Weaselboard[31] shown in Figure2.5 uses PLC backplane to get the sensor data and and actuator commands, and analyses them to prevent zero day vulnerabilities.

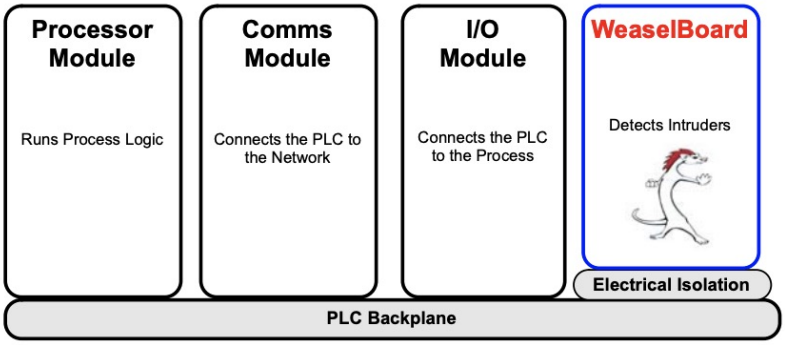

Figure 2.5: WeaselBoard in a PLC chassis, figure by Mulder[31]

In [4], model-based attack detection schemes in water distribution systems are presented. The data driven model is got from Matlab identification tool and can be used to detect process anomalies. In [18] a water control system is modeled to monitor the communication of PLCs in two real-world water plants, extracting operational semantics at the level of individual process variables. Besides water treatment system, there are also studies on power grid[28] and chemical plant[6].

Formal method is another choice in CPS intrusion detection. Signal level verification in [22] aims at discovering signal rules from the normal behaviors of the CPS. However, data with high complexity and high dimension is not suitable for this approach.

RNN is used for animaly detection in SWaT system[16]. However, in this study, only first stage of SWaT is studied due to the expensive training time. Only 10 attack scenarios are

used for the evaluation. In a follow-up work, the DNN and the one-class SVM models have been applied for anomaly detection in the SWaT system[20]. Figure2.6 shows the DNN architecture. In this work all stages and attack scenarios are considered in this work and detailed evaluation is given out. We use this literature to compare with our purposed models.

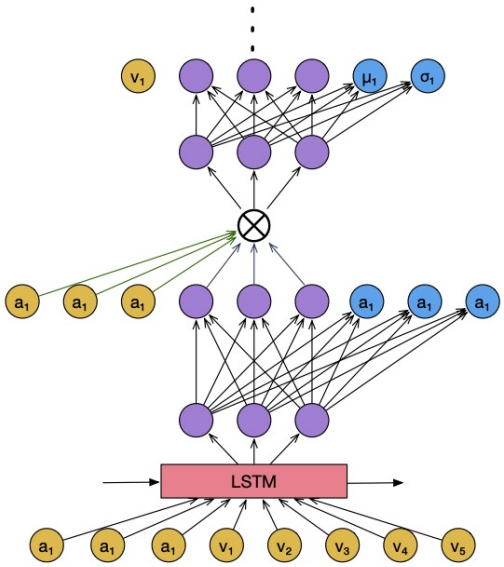

Figure 2.6: DNN architecture, figure by Inoue[20]

In [29, 32, 45], Timed automaton is learned and used for the anomaly detection. However, signals from sensor and actuator are mixed up in these study, which leads to a states blow-up problem and difficulty of localize the anomalies.

Chapter 3

Data Preprocessing

In this chapter, we will try to answer research question 1 mentioned in Chapter 1. We will first introduce the testbed, SWaT system and the dataset we use in this research. Then we will present the procedure of data preprocessing. The emphasis will be laid on the difference compared with preprocessing work for offline version. We will also present some further study and relevant discussion on this topic.

3.1 SWaT

In this research, we use SWaT system as our testbed and research object. SWaT is a fully operational scaled down water treatment plant. This testbed replicated large modern plants for water treatment which are found in cities[1]. Figure3.1 is a picture taken from SWaT Testbed. The goal of this testbed design is to research the secure problems in industrial control system. System's response to attacks can be investigated in this testbed and collected data can be used in the design of secure strategy.

SWaT system has six sub-processes, referred to as six stages,which are controlled by independent Programmable Logic Controllers(PLC). Figure3.2 shows the detailed process of water treatment in SWaT system. Using data from sensors and actuators, the system can estimate the current state and give out control actions.

In order to help understand SWaT system, we give out a brief introduction of six sub-process here.

- Stage P1: The main function of P1 is controlling the inflow of water to be treated by opening or closing a motorized valve.
- Stage P2: The water tanks used in P2 is chemical dosing stations which accepts raw water from P1 and does chlorination.
- Stage P3: In stage P3, chlorinated water is pumped to another Ultra Filtration(UF) feed water tank.

Figure 3.1: SWaT Testbed

- Stage P4: In stage P4, an Reverse Osmosis(RO) feed pump sends water through an ultraviolet dechlorination unit controlled by a PLC. The goal of this step is to remove any free chlorine from the water prior to passing it through the reverse osmosis unit in stage P5. Sodium bi-sulphate (NaHSO3) can be added in stage P4 to control the Oxidation Reduction Potential.
- Stage P5: In stage P5, the dechlorinated water is passed through a 2-stage RO filtration unit. The filtered water from the RO unit is stored in the permeate tank and the reject in the UF backwash tank.
- Stage P6: Stage P6 controls the cleaning of the membranes in the UF unit by turning on or off the UF backwash pump.

Another structure needs to be introduced is the backwash circle in SWaT system. The backwash cycle is initiated automatically once every 30 minutes and takes less than a minute to complete. A backwash cycle is also initiated if the pressure drop exceeds 0.4 bar which indicates that the membranes in the UF unit are clogged and need to be cleaned.

Besides functions mentioned above, a more detailed introduction about SWaT architecture can be found in Adepu's paper[1].

3.2 Dataset

The dataset we use in this study was collected over 11 days of continuous operation. Some detailed information of dataset can be found in Table3.1.

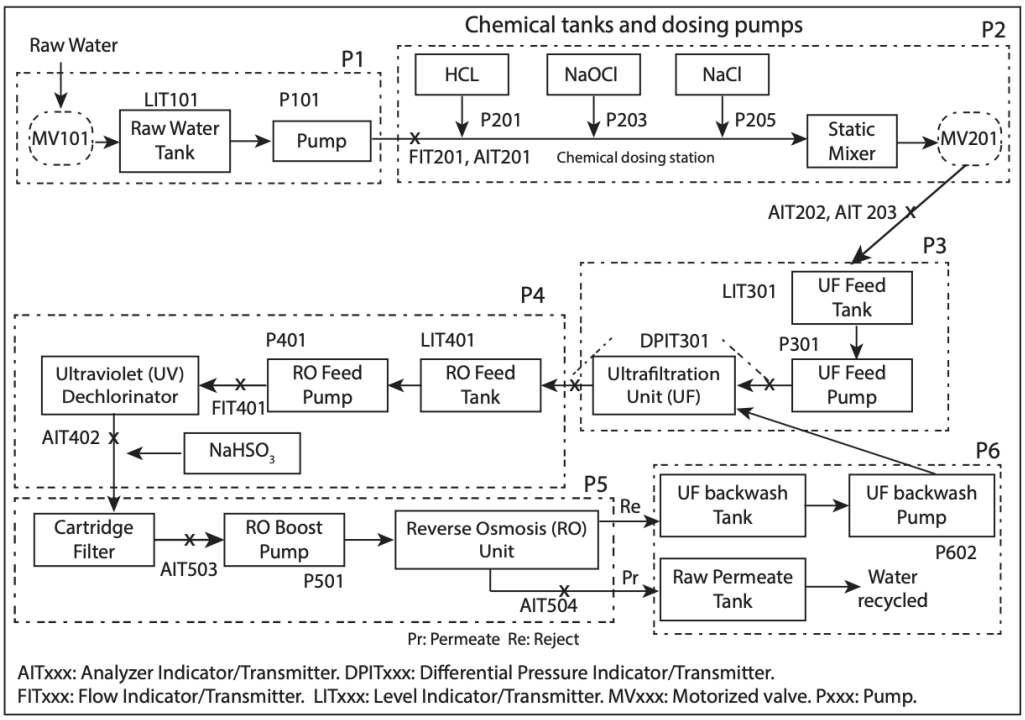

Figure 3.2: Water treatment in SWaT, figure by Adepu[1]

Table 3.1: Information about SWat Dataset

Time	22/12/2015 4:00:00 PM - 2/1/2016 2:59:59 PM
Sampling Time	1 second
Size	946719 rows * 53 columns
Training Dataset	22/12/2015 4:00:00 PM - 28/12/2015 9:59:59 AM
Testing Dataset	28/12/2015 10:00:00 PM - 2/1/2016 2:59:59 PM
Number of Attack Scenarios	36
Attack Duration	101 seconds - 10 hours

Among this dataset, the first 7 days of data and remaining 4 days of data are separated as training and testing dataset. In the first 7 days, the SWaT system is under normal operation, so the data collected is without any attacks. In testing dataset, 36 designed attack were simulated on SWaT testbed and collected.

Each data point in SWaT dataset has 53 different values. Among them, one is for timestamp, one is for label('Attack' or 'Normal'), and the remaining 51 are recorded data of 51 sensors and actuators.

The design of attack scenarios is worth introducing. All the 36 attack scenarios are generated based on attacks reported earlier[1, 2]. The duration of these attack varies from 101 seconds to 10 hours with the type and intent of attack. Some attacks are launched following the last attack within 10-minute gap. In other case, there is enough time for the system to stabilize after some attacks.

All 36 attack scenarios can be divided into four types according to numbers of attack points and stages included: 26 Single Stage Single Point (SSSP) attacks performed on exactly one point in cyber physical system; 4 Single Stage Multi Point (SSMP) attacks on two or more points but on only one stage; 2 Multi Stage Single Point (MSSP) attacks and 4 Multi Stage Multi Point (MSMP) attacks performed on two or more stages.

All the attacks are performed by injecting the process variable values into Programmable Logic Controllers (PLCs). In this way, PLCs will be leaded into believing the spoofed value.

Among all attacks, some attacks are launched randomly on random sensor, while there is also some attacks that simulates stealthy attacker's behaviour and changes the value slowly.

The detailed description of 36 attack scenarios can be found in Table3.2.
Here we give out an example of attack. Figure3.3 shows the procedure of attack scenarios 8 and 9.

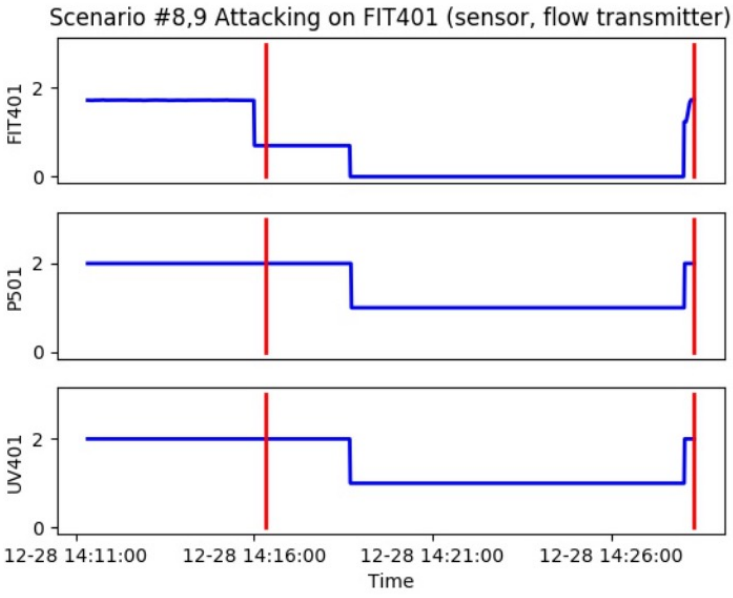

Figure 3.3: Attack Example, figure by Lin[26]

In this attack, the value of FIT-401 are injected with 0.7 and then 0 while the real value is

Table 3.2: 36 attack scenarios in attack dataset

NO.	NO. Scenario	Description of attack
1	1	Open MV-101
2	2	Turn on P-102
3	3	Increase LIT-101 by 1mm every second
4	4	Open MV-504
5	6	Set value of AIT-202 as 6
6	7	Water level LIT-301 increased above HH
7	8	Set value of DPIT as >40kpa
8	10	Set value of FIT-401 as <0.7
9	11	Set value of FIT-401 as 0
10	13	Close MV-304
11	14	Do not let MV-303 open
12	16	Decrease water level LIT-301 by 1mm each second
13	17	Do not let MV-303 open
14	19	Set value of AIT-504 to 16 uS/cm
15	20	Set value of AIT-504 to 255 uS/cm
16	21	Keep MV-101 on continuously; Value of LIT-101 set as 700mm
17	22	Stop UV-401; Value of AIT-502 set as 150; Force P-501 to remain on
18	23	Value of DPIT-301 set to >0.4 bar; Keep MV-302 open; Keep P-602 closed
19	24	Turn off P-203 and P-205
20	25	Set value of LIT-401 as 1000; P402 is kept on
21	26	P-101 is turned on continuously; Set value of LIT-301 as 801mm
22	27	Keep P-302 on continuously; Value of LIT-401 set as 600mm till 1:26:01
23	28	Close P-302
24	29	Turn on P-201; Turn on P-203; Turn on P-205
25	30	Turn P-101, MV-101 on; LIT-101=700mm; P-102 starts
26	31	Set LIT-401 to less than L
27	32	Set LIT-301 to above HH
28	33	Set LIT-101 to above H
29	34	Turn P-101 off
30	35	Turn P-101 off; Keep P-102 off
31	36	Set LIT-101 to less than LL
32	37	Close P-501; Set value of FIT-502 to 1.29 at 11:18:36
33	38	Set value of AIT-402 as 260; Set value of AIT-502 to 260
34	39	Set value of FIT-401 as 0.5; Set value of AIT-502 as 140 mV
35	40	Set value of FIT-401 as 0
36	41	Decrease LIT-301 value by 0.5mm per second

around 2. This value shows the water flow form stage 4 to stage 5. When system believes

that there is no water transmitted from stage 4 to stage 5, the pump between two stages will be turned off. In this case, the water in stage 4 can not transmitted to next part of system anymore. And finally it leads to overflow in water tanks in stage 4.

3.3 Data preprocessing

3.3.1 Preliminary Study of Dataset

In previous introduction, We mention that SWaT system has six sub-processes, referred to as six stages. After preliminary study of SWaT dataset, we can find that data of several sensors and actuators in stage 6 is not collected in dataset. On the other hand, there is also no attack launched on stage 6 in testing dataset. So our study do not contain sensors and actuators in stage 6.

Besides stage 6, we can split data according to stages. We split 16 sensors into 16 sub-models. Each sub-model contains one sensor and all the actuators in the same stage as that sensor. The split result is shown in Table3.3.

Table 3.3: Sub-model Split with Sensors and Stages

NO.	Stage	Sensor	actuator
1	1	LIT101	MV101, FIT101, P101, P102
2	2	AIT201	FIT201, MV201, P201, P203, P205
3	2	AIT202	FIT201, MV201, P201, P203, P205
4	2	AIT203	FIT201, MV201, P201, P203, P205
5	3	DPIT301	FIT301, MV301, MV302, MV303, MV304, P302
6	3	LIT301	FIT301, MV301, MV302, MV303, MV304, P302
7	4	AIT401	FIT401, P402, P403, UV401
8	4	AIT402	FIT401, P402, P403, UV401
9	4	LIT401	FIT401, P402, P403, UV401
10	5	AIT501	FIT501, FIT502, FIT503, FIT504, P501
11	5	AIT502	FIT501, FIT502, FIT503, FIT504, P501
12	5	AIT503	FIT501, FIT502, FIT503, FIT504, P501
13	5	AIT504	FIT501, FIT502, FIT503, FIT504, P501
14	5	PIT501	FIT501, FIT502, FIT503, FIT504, P501
15	5	PIT502	FIT501, FIT502, FIT503, FIT504, P501
16	5	PIT503	FIT501, FIT502, FIT503, FIT504, P501

As we can see in the table, sensors and actuators are named with type label and numbers. Among these three-digit number, the first number shows the stage in which sensors and actuators locates, while the other two shows index of sensors and actuators in each stage.

All the actuators in SWaT dataset can be considered to have two states. In our study, these two states can be represented by two numbers, 1 and 2. Value 1 means that the actuator is closed, while value 2 means that the actuator is open.

SWaT consists of an array of monitoring sensors to ensure its safe operations. Sensors with different labels measure different indexes in SWaT system:

- LITxxx Sensors: Level Indication Transmitter measures water levels in mm.
- AITxxx Sensors: Analyser Indicator Transmitter measures conductivity(μS/cm), pH or Oxidation Reduction Potential(mV).
- DPITxxx Sensors: Differential Pressure Indicator Transmitter measures differential pressure in kPa.
- PITxxx Sensors: Pressure Indicator Transmitter measures pressure in kPa.

Among all four types of sensors, LITxxx and DPITxxx sensors show sequential measure, while AITxxx and PITxxx sensors includes large noise and show subtle trends which is hard to model. According to these facts, the emphasis of modelling should be laid on LITxxx and DPITxxx sensors, in other words, sub-model 1, 5, 6 and 9.

3.3.2 Denoising

According to the design of SWaT, there is a hard filter in each stage[2]. All the data collected has been denoised already. On the other hand, the sampling frequency of dataset we use in this study is once per second, which is not quite high compared with other cyber physical system dataset. However, we can find that there are still many pikes in sensor data sequence. This is mainly because of the environment of data collection.

In supervised model learning, the quality of training data has a great influence on model's final performance. These pikes in dataset can arouse big problem in the following learning procedure. So denoising is needed before other data preprocessing procedure.

To present the procedure of denoising, we use one-dimensional time series as example. We define a one-dimensional time series of a sensor signal is defined as:

$$x[n] = [x_1, x_2, x_3, \cdot, x_n]$$

We simply use a naive averaging filter to denoise the signal in this study. The denoised time series is defined as:

$$\bar{x}[w] = [\bar{x}_1, \bar{x}_2, \bar{x}_3, \cdot, \bar{x}_w]$$

where the ith element of $\bar{x}$ can be calculated by:

$$\bar{x}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i}$$

The above procedure can be easily applied on training dataset. And there is no difference whether we apply the normalization before or after denoising as we use a linear filter. Figure3.4 shows an example of original and denoised signal.

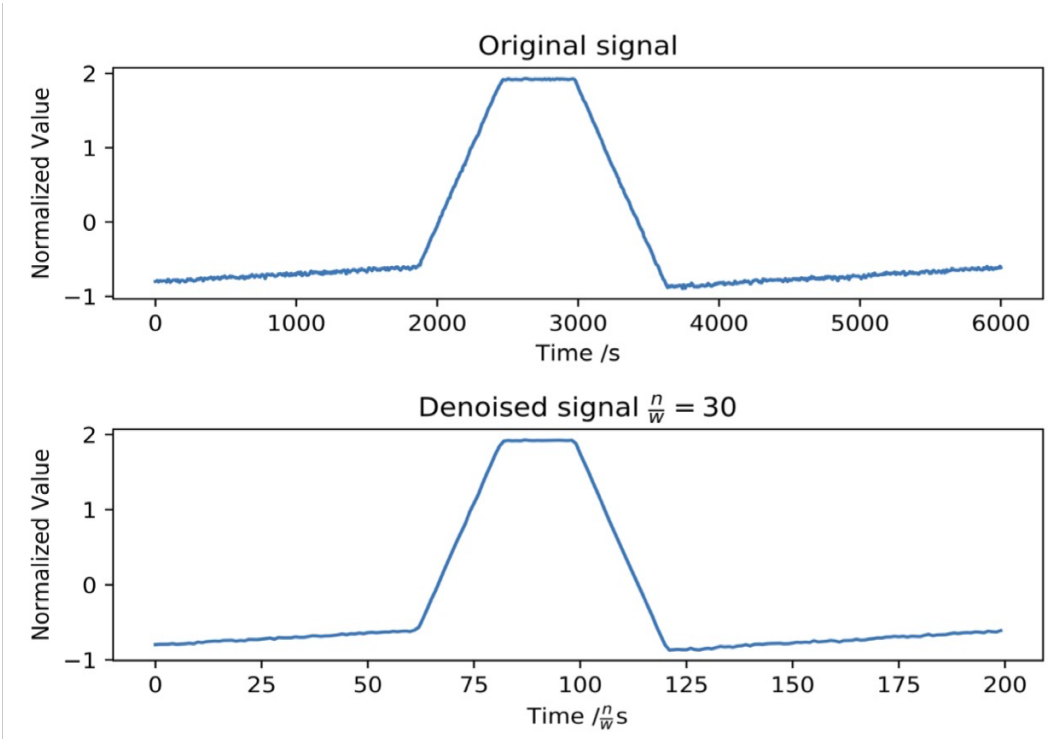

Figure 3.4: Example of Original and Denoised Signal

In real-time testing, we get an average data point each time we receive another $\frac{n}{w}$ data point and use the configuration calculated from training dataset to do the normalization.

There are several details need to be discussed here:

- Value of $\frac{n}{w}$: In example shown in figure, $\frac{n}{w} = 30$, which means we averages every 30 orignal data points into a denoised data point. This procedure will change the sampling frequency of dataset in some way. The bigger the value is, the lower the sampling frequency is. In Qin's study, he just uses 60. However, in this study we adjust this value based on the result of following preprocessing and modelling procedures and try to find minimum acceptable value to increase the sampling frequency. We think this is very necessary because if smaller value can be found, the frequency

of alarms will correspondingly increases, which means the system responds to incoming data more quickly. Finally, we choose 30 in sub-model 9, while in sub-model 1, 5 and 6, the choice of 30 arouse bad result of signal segmentation. So we still have to use 60 in that three sub-models. We also find out that the acceptable minimum value of $\frac{n}{w}$ is highly related to the quality of dataset used in study, so we believe with this new procedure included and a better dataset, our proposed model can respond to data and give out alarms more quickly.

- Divisibility of n: For simplicity and clarity, we assume that n is divisible by w in above discussion. In training part, if n is not divisible by w, we can easily give up the remainders in x. However, in testing we have to give out the detection result for the remainders of x, so we can simply add an additional chunk appending to $\bar{x}$ averaging the remainders in x. In this case, we can also find that a smaller $\frac{n}{w}$ can help arouse less noise in end point of testing.

3.3.3 Segmentation

By denoising and normalizing sensor signals, the sequential behaviour becomes more obvious. We still need some techniques to represent the denoised sensor signal. A good design of representation is essential to efficient and effective solutions to time series modelling.

When it comes to time series representation, piecewise linear representation(PLR) is one of widly used method, which supports clustering, classification, indexing, and association rule mining of time series data. In our study, we extract the basic idea in PLR and design our segmentation algorithm based on difference between data points. We set max error as threshold for the segmentation. Two different methods of using difference are tested:

- We calculate the difference between every pair of neighbouring data points and compare it with max error.
- We calculate the difference between data point and the whole chunk without any segmentation before that point.

After several attempts, we find that more reasonable result can be got by using the second method. The pseudo code of segmentation algorithm is shown as Algorithm1.
An example of segmentation is shown in Figure3.5. In this segmentation of LIT101 sensor data, we use 0.2 as max error ξ and 60 as $\frac{n}{w}$. The choice of these two values are studied from a large number of attempts. However, we fail to find some efficient method to judge the quality of segmentation, so we use visual check here.

In training part, similar segmentation are applied on data of all four sensors used in sub-model 1, 5, 6 and 9. In real-time testing, we need extra data space to record the chunk temporarily before each segment happens to realize the comparison between newly received data point and the chunk before.

Algorithm 1 Segmentation Algorithm

Require: Normalized time series $\bar{x}$, max error ξ
Ensure: Set of Segmentation *Seg*
  $index = 1$
  $chunk_mean = 0$
  $Seg = \oslash$
  while time series does not end **do**
    $i = 2$
    while $|\bar{x}_i - \bar{x}_{i-1}| \leq \xi$ **do**
      $i = i + 1$
      $chunk\_mean = update\_chunk\_mean(\bar{x}[index : index + (i-1)])$
    end while
    $Seg = concat(Seg, create_segment(\bar{x}[index : index + (i-1)]))$
    $index = index + 1$
  end while

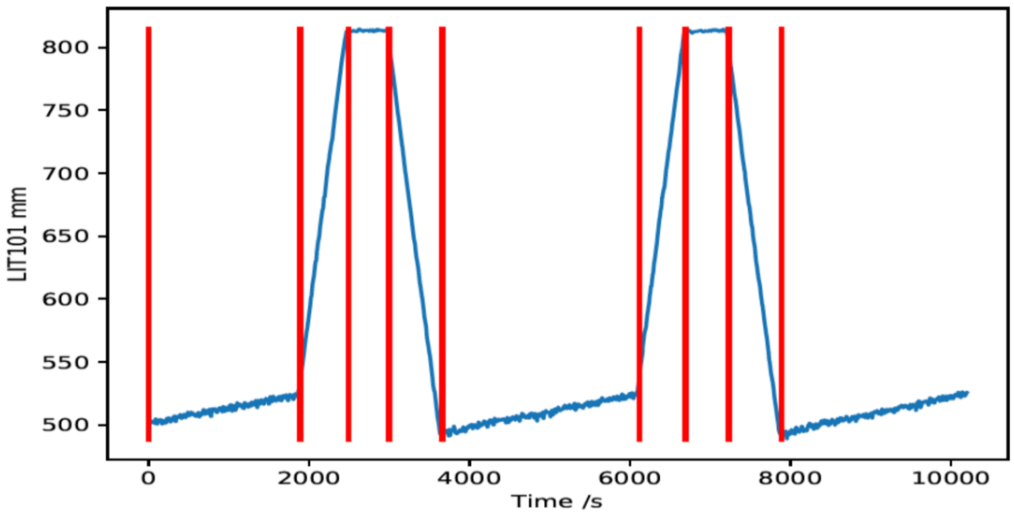

Figure 3.5: Example of Segmentation on LIT101 Data

3.3.4 Representation

After getting the segment of sensor data, clustering algorithm is needed for discretization and representation. The inputs to the clustering algorithm are the differential values of the segments *chunk_mean* obtained in the segmentation step.

We try two different clustering methods here, k-means method and k-quantile clustering algorithm. However, when choosing the same value for k, the results of two clustering methods are quite the same. After several attempts, we decide suitable cluster number for each sub-model. We use Figure3.5 as our example again. According to the data study, we

cluster data of LIT101 into 4 different group. This conclusion can also be confirmed by the figure. As we can see, four different trends are obvious. We name these four trends as quick down(QD), staying constant(SC), slow up(SU) and quick up(QU). In following modelling, four numbers, 1, 2, 3 and 4 are used to represent these four symbolic events given out by clustering.

In real-time testing, once we get a denoised data point, we decide the symbolic representation for it according to recorded index got from training data representation.

3.3.5 Alignment

Once we finish the preprocessing on sensor data, we use the timestamp got from sensor signal's segments to obtain the corresponding status of actuators.

As all the actuator data in SWaT dataset can be considered to have two states(1 - closed, 2 - open), no extra preprocessing is needed. Figure3.6 shows an example of alignment of the sensors and the actuators in stage 1.

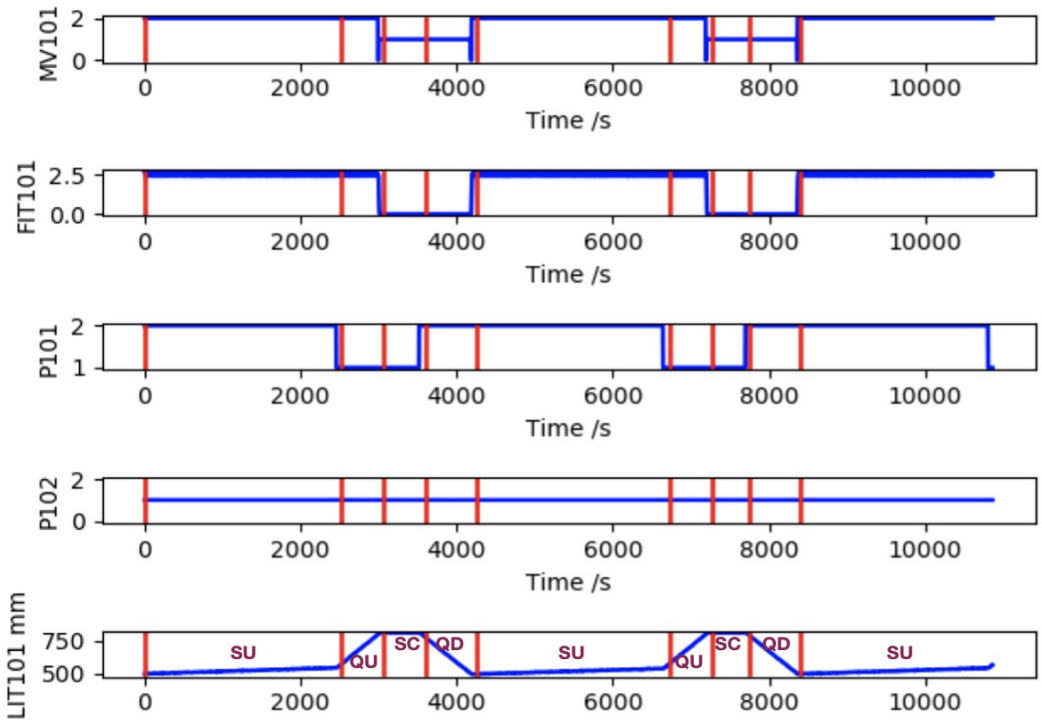

Figure 3.6: Example of Alignment, figure by Lin[26]

In this example, a time strings of sensor is used:

$(SU, 2520)(QU, 540)(SC, 540)(QD, 660)(SU, 2460)(QU, 540)(SC, 480)(QD, 660)(SU, 2460)$
Each bracket corresponds to a chunk from segmentation result. The numbers means the duration of chunks. Using timestamp, we can easily mark every segment on actuator data strings. However, it is obvious that there is some misplacement, especially near the segments, which is mainly aroused by noise in original signal.

To decide the corresponding state for every chunks, we try two different methods here:

- Mid point value: We pick the mid point of each chunk and use its value as the value of the whole actuator data chunk.
- Majority vote: We apply majority vote on each chunk and decide its value from the result.

According to the result, when we use 30 or 60 as $\frac{n}{w}$ in denoising, the alignments from these two methods are quite the same. This is mainly because the quality of actuator data is good enough. However, with simple check, we can find that actuator data in SWaT dataset contains some pikes in time string, mostly happens near the transition. Although in this study, proposed systems perform not well when $\frac{n}{w}$ is smaller than 30, which is mainly related with the quality of dataset, in real world problem we often make an effort to increase sampling frequency of dataset. So we also test the alignments when the value of $\frac{n}{w}$ is 5 or 10. In these cases, we find that mid point value method arouses more noise compared with majority vote, which means in high sampling case majority vote method has a better performance. So we recommend majority vote here as mid point value. Majority vote also performs better when quality of data can not be guaranteed.

In real-time testing, we receive testing data continuously. Although we can accurately find the segments by comparing difference, problems still happens in alignments. By studying the dataset, we can find that in most cases, the misplacement of segments in actuator data is that the sensor data have a delay compared with actuator data, which we can also find in FIgure3.6. However, in some cases, things happens in a contrary. For example, assume we have a time strings of sensor and actuator data:

- Sensor strings: (SU, 2520) (QU, 540) (SC, 540) (QD, 660)
- Actuator strings: (Open, 2580) (Closed, 480) (Open, 540) (Closed, 660)

We use 60 as $\frac{n}{w}$ in this example, which means we deal with data of 60 timestamps each time. By apply segmentation on sensor data, we can first get a segment on timestamp 2520. Then problem happens in timestamp 2520-2580. According to the logic of alignment, we will first consider it as open first and find it should be actually considered as closed. This kind of noise can be considered as a kind of physical delay and can influence the result of detection.

We also find that this kind of problem hardly arises when $\frac{n}{w} \geq 60$ and gradually shows up with the decrease of $\frac{n}{w}$(increase of sampling frequency). As is mentioned above, we use 30 in sub-model, so extra design is needed in modelling to minimize the influence of this kind of noise. We will discuss it with an example in next Chapter.

Chapter 4

Real-time System

In this chapter, we will present the real-time version detection system to answer the research question 2. First, we will briefly introduce the Out of Alphabet and its performance. Secondly, we will discuss about Time Automaton and some relevant study about the difference between offline TABOR model and real-time model. Then we will study the Bayesian Network part and discuss the difference in real-time modelling and testing. Finally, we will give out an evaluation of real-time system we implement.

4.1 Out of Alphabet

The basic idea of Out of Alphabet(OOA) is quite simple. In OOA method, training dataset is used to form a set of values or ranges for normal data. Then every testing data which is not contained in the set or exceeds the range will be marked as anomaly.

The cost of OOA is quite low. However, it tends to show obviously abnormal patterns with a high priority in our study. There are two main reasons:

- The data of AITxxx and PITxxx sensors shows no obvious sequential behaviour, which make it hard to model them. OOA is a good choice considering its flexibility and low cost.
- Among all attack scenarios against industrial control system, a large number of attacks are launched by injecting or shifting the values directly. When dealing with this kind of attacks, OOA can find the abnormal value swiftly.

In our study, we use OOA in several different ways:

- We deploy a symbolic check on all the actuator data to find the data points which is neither ”closed” nor ”open”. Although this can hardly happen actually, we still do the check.
- We deploy a range check on all sensor data to find the data exceeds normal range.

- We calculate the deviation of the differential of AITxxx and PITxxx sensors and form a range of normal deviation range. And then a range check is deployed on testing data. As we can not study the sequential behaviour of these two kinds of sensors, we use this method to check their changes along with time.

Two example of detection result from AIT202 and PIT510 are shown in Figure4.1 and Figure4.2.

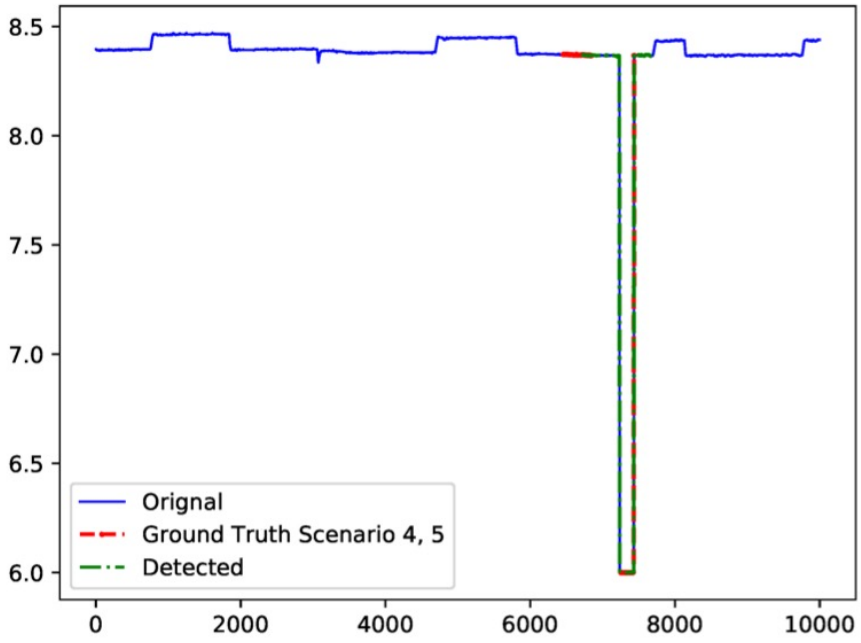

Figure 4.1: An example of detection result from AIT202

Both examples here are detected with OOA check on deviation of differential. This kind of value injecting attack can easily find with the help of OOA method.

To further study the performance of our OOA design, we deploy a real-time testing with OOA-only strategy. The detection result is shown in Table4.1.
The number in table means model's recall in each detected attack scenarios. We can find that OOA-only strategy successfully detects 12 attack scenarios out of all 36 attack scenarios. Moreover, in most detected scenarios, the recall is quite close to 1. From literature we know that SVM-based and DNN-based model detect 13 and 20 scenarios. Compared with these two model, we can say that OOA-only strategy shows good performance with low cost.

According to the above study, we decide to adopt the OOA errors directly into the final result. And in our following study, we only consider the modelling and testing without

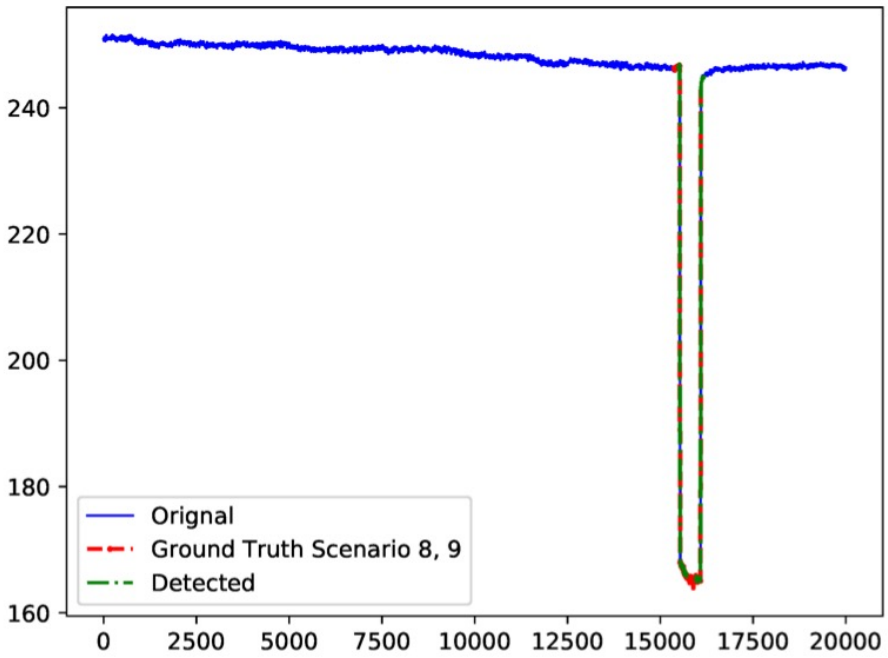

Figure 4.2: An example of detection result from PIT501

Table 4.1: Points evaluation of OOA-only Strategy

Model	OOA-only
Scenario 4	0.328
Scenario 5	0.995
Scenario 8	0.994
Scenario 9	0.998
Scenario 14	0.004
Scenario 15	0.997
Scenario 17	0.998
Scenario 23	1.000
Scenario 32	0.998
Scenario 33	0.996
Scenario 34	0.369
Scenario 35	0.997

OOA. When it comes to the final model, we means the combination of OOA and proposed model with ”OR” strategy.

4.2 Timed Automaton

4.2.1 Probabilistic Deterministic Real Timed Automaton

Based on denoising, segmentation and representation, regular symbolic time string can be represented by Deterministic Finite Automata(DFA), which are also common models for discrete event. The DFA is defined in Definition4.2.1.

Definition 4.2.1. A DFA 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($\delta = Q \times \Sigma \to Q$) 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 Figure4.3 in the format of state diagram. S_0, S_1, S_2 are three states of this DFA and S_0 is the start state. A finite sequence of 0 and 1 are the input of this automaton. When reading a symbol, a DFA moves deterministically from one state to another, followed the arrow.

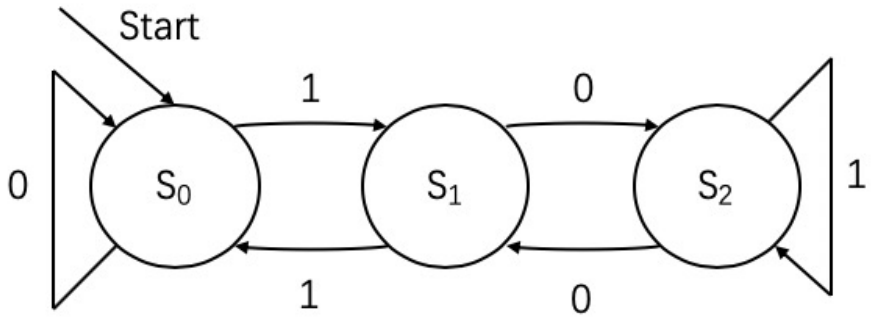

Figure 4.3: An example of DFA

There is an variant of DFA, named as probabilistic deterministic finite automaton(PDFA). The difference between DFA and PDRTA is that PDFA records the probability of the emitted sypmbol given a state. The PDFA is defined in Definition4.2.2. The probability of a sequence can be easily obtained by multiplying all the state-symbol probabilities along such a path. A PDFA example is demonstrated in Figure4.3.

Definition 4.2.2. A PDFA is a 5-tuple $\langle Q, \Sigma, \delta, \pi, q_0 \rangle$, where Q is a finite set of states, Σ is a finite set of input symbols called the alphabet, δ is a transition function($\delta = Q \times \Sigma \to Q$) representing one state/symbol to the next, $\pi : Q \times \Sigma \to [0,1]$ is the probability of the emitted symbol given a state, and q_0 is an initial or state belonging to Q.

However, the data obtained from real world is not always only in the type of discrete event sequence. In practical world, events always happen with timing, such as our dataset, which

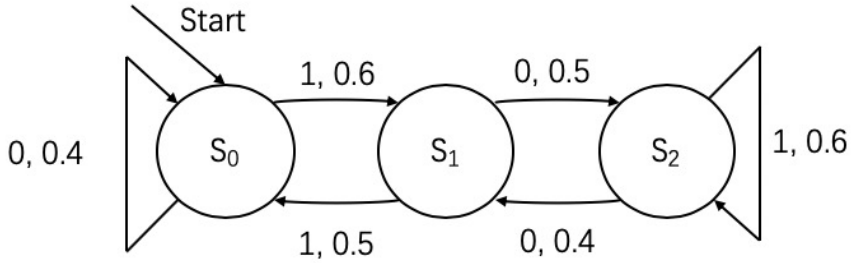

Figure 4.4: An example of PDFA

contains time values of event transitions, and DFA and PDFA has limited capability of handling this obstacle. As these time values are considered as a feature in our study, we introduce time automaton, Probabilistic Deterministic Real-Time Automaton(PDRTA). An algorithm for efficient learning of timed automata algorithm is proposed in[40, 41]. In our research, we represent time series into the format $(a_1,t_1),(a_2,t_2),\cdots,(a_n,t_n)$ to learn the PDRTA model, where a_i is a discrete event and t_i is the duration of each event a_i. The PDRTA is dinfined in Definition4.2.3.

Definition 4.2.3. A PDRTA is a 4-tuple $\langle A,H,S,T\rangle$, where $A=\langle Q,\Sigma,\Delta,q_0\rangle$ is 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 $S_q = Pr(S=a|q)|a\in\Sigma,q\in Q$, and T is a finite set of time-bin probability distributions $T_q = Pr(T\in h|q)|h\in H,q\in Q$.

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

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

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

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

Figure4.5 shows an example of PDRTA[40]. 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. The PDRTA can be used as prdictor of timed events. Let $H=\{[0,2];[3,4];[5,6];[7,10]\}$ be the histigram. In every bin the distribution over time values is uniform. For example, the probability of $(a,3)(b,1)(a,9)(b,5)$ is

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

.

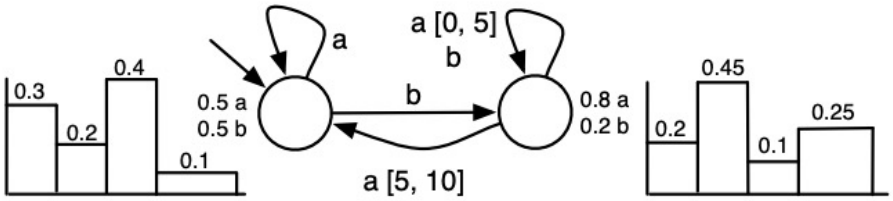

Figure 4.5: An example of PDRTA, figure by Verwer[40]

In PDFA and PDRTA, the states are latent variables that cannot be directly observed in strings, but have to be estimated by using a learning method. As a timed variant of PDFA, PDRTA is trigged when both an event and its timing are validated. Therefore, we use PDRTA in this study and choose corresponding learning method for it.

4.2.2 TA learning

RTI+ is a state-of-the-art machine learning algorithm. It can be used to learn human behaviors from unlabeled data[42]. We use RTI+ in our research to model TA.

In traditional state machine learning, a large tree-shaped automaton called augmented prefix tree acceptor(APTA) is built from all input strings. All the states of APTA can be reached by exactly on string. When it comes to timed learning, we add time duration in APTA and get a timed APTA(TAPTA). The lower and upper bounds of each time duration are decided by the minimum and maximum time values from all input strings. Figure4.6 shows a TAPTA built from timed strings. The input strings sample used in this TAPTA is:

$$S = (a,1),(a,1)(b,2)(b,1),(b,2)(b,1),(a,1)(b,1)(a,1),(b,2),(b,1)(b,1)$$

In our study, our dataset can be represented into a long time seires. However, we can not input the whole series in one time to built the TAPTA, So we rewrite the data strings into a large number of short sentences. One sentence consists of two full cycles to better capture any looping behaviour in the state machine. For example, in sub-model 1, we have a sentence:

$$(3,1620)(2,660)(1,660)(3,840)(4,600)(2,780)(3,2760)$$

Each element in time series sentence consists of symbolic event and time duration. And the sentence contains two full cycles which start and end with symbolic event 3. And with these sentences, algorithm can built TAPTA for our model.

After TAPTA is built from timed strings, state merge and transition splits are two main operations.

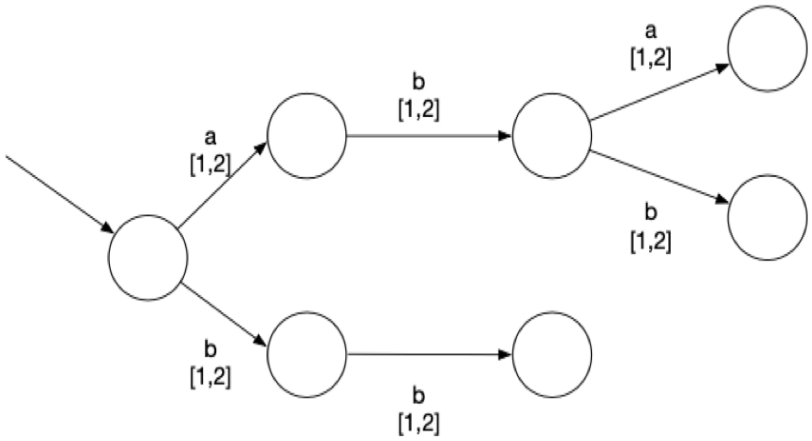

Figure 4.6: An example of TAPTA, figure by Lin [26]

A split of a transition $\langle q, q', a, [v, v'] \rangle$ at time t creates two new transitions $\langle q, q_1, a, [v, t] \rangle$ and $\langle q, q_2, a, [t+1, v'] \rangle$. The target states q_1 and q_2 are the roots of two new prefix trees which are reconstructed based on input series.

A merging of states merges pairs of states (q, q'), forming an smaller machine, and all the states that are reached from q to q' have to be merged as well.

In RTI+, the algorithm greedily search for every possible state merge and transition splits to find the final structure of TAPTA. So a method is needed to decide whether to split/merge or not. The method we use here is a likehood-ratio statistical test[40]. 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'. According to the definition, H' has more unconstrained parameters than $H(n' > n)$. In our study, the model after merge or before split is H and the model before merge or after split is H'. Given two hypotheses H and H' such that 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. The likelihood-ratio test can test whether this increase in likelihood is significant or not. The test compares the value $-2ln(LR)$ to a $\chi^2(n' - n)$ distribution. The result of this comparison is p-value. When p-value is high, it indicates that H is a better model. We use a 95% confidence level here, which means

- When p-value is less than 0.05, the split is accepted.
- When p-value is greater than 0.05, the merge is accepted.

Each time we perform the split with the least p-value or the merge with the highest p-value. An overview of RTI+ is shown in Algorithm2.

Algorithm 2 RTI+ Algorithm

Require: A multi-set of time string S_+
Ensure: A small PDRTA A for S_+
  Construct a timed prefix tree from S_+
  $Q' = \oslash$
  for all transitions $\langle q, q', a, [v, v'] \rangle$ from A **do**
    Evaluate all possible merges of q' with states from Q'
    Evaluate all possible splits
    if the lowest p-value of splits < 0.05 **then**
      Perform this split
    else if the highest p-value of splits > 0.05 **then**
      Perform this merge
    else
      add q to Q'
    end if
  end for

4.2.3 Model Analysis

The sub-model 1, learned from sensor LIT101, is shown in Figure4.7. 1, 2, 3 and 4 are symbolic events Quick Down(QD), Stay Constant(SC), Slow Up(SU) and Quick Up(QU).

Figure 4.7: Timed automaton learned from sensor LIT101

In this model, there are two main components, state and transition condition. States are marked in the circles, and conditions for state transition are written over the arrow. The

condition consist of four elements. The first one is time duration of the event. The second one is symbolic event. The third and fourth one are probability of the emitted symbol given a state and number of times transition occurs in training set.

The starting state is from the top as S0. From S0, it is possible to move to three other states, which are S1, S3, S. S is the sink state, which is introduced due to fact that some sequences in the training data have very low frequencies of occurrence. The sink states are left without split or merge with other states due to lack of evidence for statistical testing. In our study, we consider these kind of events are not anomalies as they actually occur in training dataset.

Starting from S0, the transition condition all limit the alphabet to be "3" and varies with time duration. Event "3" with valid time duration will make system's state move forward. If the event is not "3" or the time duration does not follow any of valid time duration, the data will be rejected and marked as anomaly. In other words, any testing sequence that is not fired by the learned TA is alarmed as an anomaly, and two main types of alarms in TA is event error(symbol that can not be fired for transition in the given state) and timing error(symbol's time duration exceeds the valid time guard).

In our study, once abnormal event is found, we consider that the abnormal event lasts until the end of the sequence sentence. The rules about sentence rewriting have been introduced above in section 1.2.2.

4.2.4 Discussion about Real-time testing

We combine 4 sub-models into final TA-only model and evaluate it on real-time operation. The results is shown in Table4.2.

Table 4.2: Detection Result from TA-only model

Testing Data Size	Precision	Recall	Detected Scenarios
450000	0.165243	0.874838	26
True Positive	False Positive	True Negative	False Negative
47124	238056	158078	6742

We can see that TA-only model is very sensitive according to the false positive explosion. So in our study, we use TA model to get the approximate range. Also another model based on Bayesian Network are used for further proof, which is introduced in next section.

The difference between real-time TA model and offline model is worth discussing about. According to our design, TA model can detect event errors(transition errors) and timing errors(transitions with invalid time duration). We discuss the difference about two kinds of errors respectively.

First, we study the difference about event errors between offline and real-time TA model. In TA-model, only sensor data series are used in modelling. So the problem about alignment which we discuss in last chapter will not influence the performance of TA in real-time operation. On the other hand, we use the same logic in series segmentation and same algorithm in modelling as offline model TABOR does. So when we only consider event errors, the detection result of real-time TA model we implement is quite the same as the offline testing result of TABOR, which can be easily confirmed by comparing both detection reports.
Then we study the difference about timing errors between offline and real-time TA model. The results of two models here are not quite the same. We use an example to explain about this. In testing dataset, we have a time series:

$$(3,2760)(4,540)(1,660)(3,2400)(4,540)(2,480)(1,660)(3,2640)$$

The timestamp of this series is from 13380 to 24060. In offline testing of TABOR, the system read a whole chunk every time. As the valid time duration of symbolic event 3 is between 0 to 2640, TABOR marks the whole chunk (3,2760) as anomaly. However, in real-time testing of our model, data is received continuously in 60s per unit. So our model do not give any alarm in first 2640s and only mark the data from 2640 to 2760 as anomaly. The same difference happens in every timing error detection. We can not conclude arbitrarily that one model is better than the other one in this case. However, we compare two results with ground truth and find out that our model arouses less false positive points while true positive points of two models are the same, which means in most cases an attack happens in the mid of a normal event other than at an edge. This result is consistent with statistical law. So we recommend this real-time strategy here.

In conclusion, we claim that our implementation on real-time TA model meet the requirements in this study. More evidence can be found in the evaluation of final combined model.

4.3 Bayesian Network

4.3.1 Bayesian Network Learning

A Bayesian network(BN) is a probabilistic graphical model. It represents a set of random variables and their conditional dependencies via a directed acyclic graph(DAG).

In our study, sensors and actuators in the same stage shows dependencies, which can be considered as a kind of system behaviour pattern. BN is learned to model these dependencies and concurrent event behaviour.

The learning of BN starts with struture learning. The method used here to learn the structure of DAG is K2 algorithm, which is widely used in BN learning[10]. The pseudo code is shown in Algorithm3. At the start, all nodes has no parents. By adding the parents relationship which brings greatest increase of structure scores, the final structure is formed. It

is worth pointing out that the sensor data is fixed as the last entry to make sure that it is not the parent node of any other variables. There are two main reasons here:

- Sensor data has much more valid values compared with actuator data. Using sensor data as parent node will undoubtedly increase the complexity of DAG.
- We use sensor data series in TA as modelling object, so we also focus on how all other variables influence sensor data variables, rather than the opposite relationship.

Algorithm 3 K2 Algorithm

Require: A set of nodes $X = (x_1, x_2, \cdots, x_n)$, an upper bound μ of parents a node can have and a dataset D
Ensure: A network structure of BN
  for all i in 1 to n **do**
    $\pi_i = \oslash$
    $P = f(i, \pi_i)$
    Flag = true
    while Flag and $|\pi_i| < \mu$ **do**
      find z in set $(X - x_i - \pi_i)$, which maximizes $f(i, \pi_i \cup \{z\})$
      $P' = f(i, \pi_i \cup \{z\})$
      if $P' > P$ **then**
        $P = P'$
        $\pi_i = \pi_i \cup \{z\}$
      else
        Flag = false
      end if
    end while
    Record parents nodes of x_i
  end for
  Draw the network structure according to parents node information

After we learn the network structure of BN, the next step of BN learning is parameter learning. This step is relatively simple. We count the probability of each node from the data and use them to obtain conditional probability distribution(CPD) tables.

4.3.2 Model Analysis

Figure4.8 shows an example of BN. This network is a sub-model learned from stage 1 dataset. Value "1" and "2" of actuator data mean "closed" and "open". And for data of sensor LIT101, 1, 2, 3 and 4 are symbolic events Quick Down(QD), Stay Constant(SC), Slow Up(SU) and Quick Up(QU). This sub-model is learned by Lin[26] for offline testing. The core idea of BN testing is to find concurrent event does not exist in training data, so an

alarm is raised if the corresponding entry in the table is equal to zero, in other words, when a zero probability event happens.

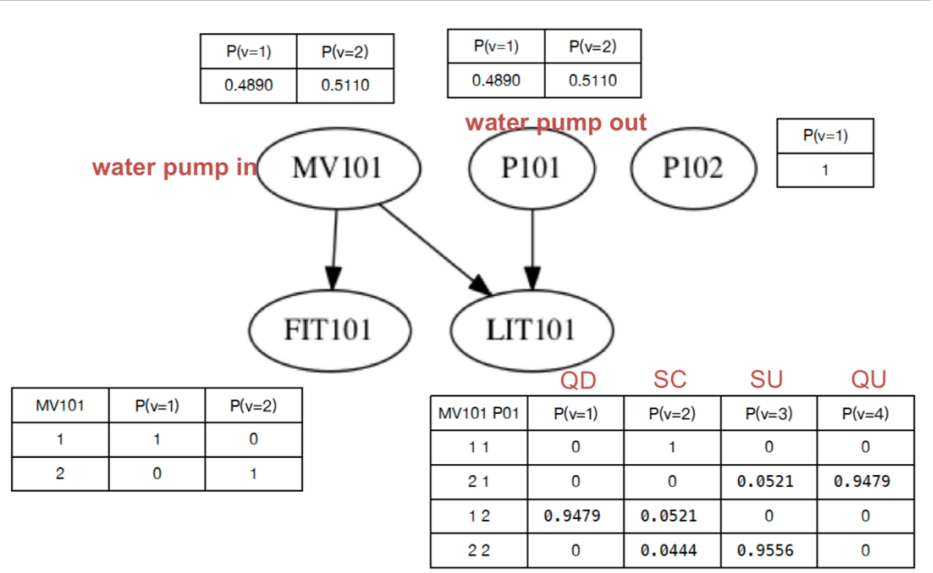

Figure 4.8: Chunk-based Bayesian Network learned from Stage 1, figure by Lin[26]

However, we find a big problem of Lin's BN model in our study about implementation of BN for real-time operation detection. As is mentioned above, the data unit used in TA learning goes through two changes. In denoising, the sampling frequency is changed, and in segmentation time series are divided into chunks because of the requirements of TA training. In BN learning, as the requirement of input is not quite strict, we have two choices of input data unit:

- We just use the denoised data points to learn a point-based BN model.
- As we have aligned the actuator data and got the state of the corresponding chunk, we can use time-series chunks to learning a chunk-based BN model.

In the BN shown in Figure4.8, chunk is chose as data unit. Actually this choice has several advantages over the other one. First, as chunk is a bigger time unit, the influence of misplacement happens in alignments can be minimized by using chunk as primary time unit. And also, unit unification of both models based on TA and BN makes the final result more convincing.

However, in real-time operation testing, problems can be aroused when using chunk as primary unit. In offline testing, as we have the whole dataset, we can easily represent the data in chunks of time series, while in real-time operation we receive data points and need to give out detection result of each points(in this case, denoised points) immediately. It means that what we get before segmentation happens can not be considered as a chunk of the same

type we used in model learning. In other words, we use different primary units in training and testing, which makes the final result unreasonable. After synthesizing all the fact, we choose denoised data points to learning a point-based BN due to the specificity of real-time operation detection.

Figure4.9 shows our point-based BN learned from Stage 1. Compared with offline model, the DAG structure and CPD are more complex. Then we evaluate our model on real-time operation.

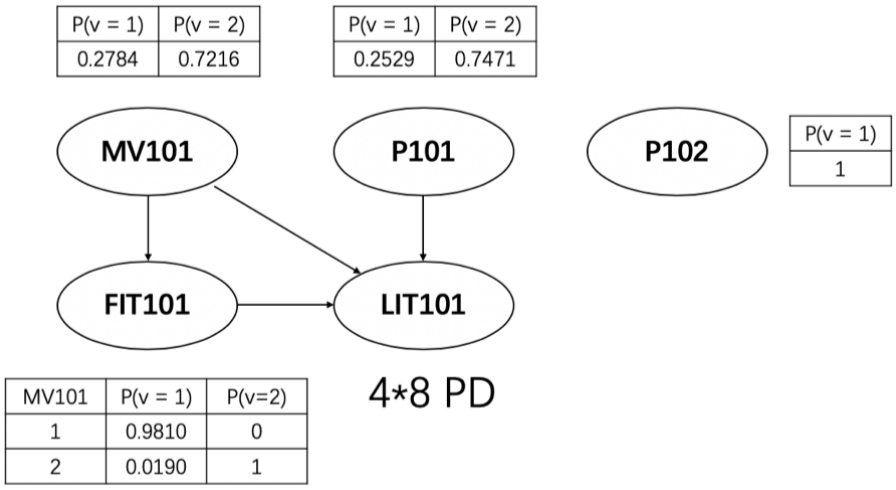

Figure 4.9: Point-based Bayesian Network learned from Stage 1

4.3.3 Discussion about Real-time testing

We still use the sub-model learned from stage 1 as example.

We first choose the same detection strategy of offline model that an alarm is raised if the corresponding entry in the table is equal to zero. However, in that case, our model give out a large number of false positive points. Comparing our model with offline model, we can find the greatest difference happens in the relationship between MV101 and FIT101. In offline model, MV101 and FIT101 shows synchronous change, which means The values of these two actuators are fully dependent. And in our model, small probability events happen which destroy this fully dependent relationship.

To find the fixing method for this problem, we first further study the functions of sensors and actuators. Figure4.10 shows the structure of stage 1 in SWaT.

According to system design, MV101 is the actuator which controls whether to pump in raw water or not. And FIT101 is used to monitor whether there is water pumped in. So in the

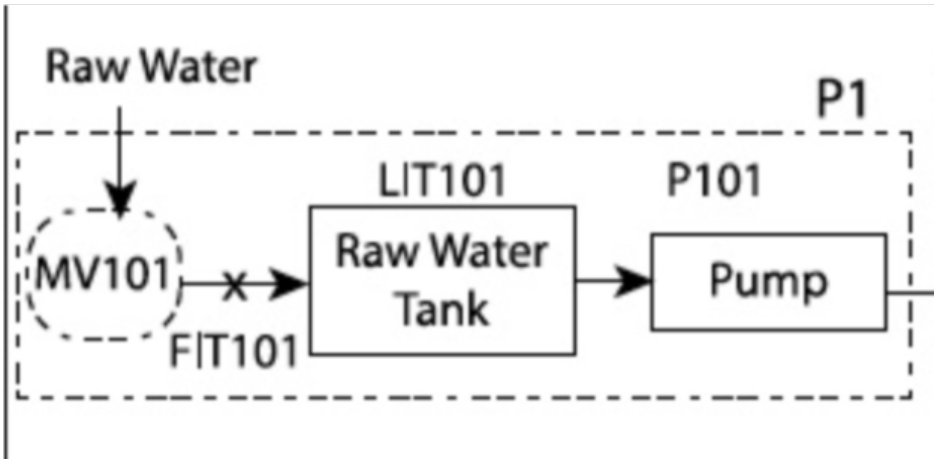

Figure 4.10: Structure of Stage 1

logic of design, data from them should shows synchronous change. However, in real world operation, the physical delay happens in this part. After MV101 is switched on(or off), it takes FIT101 some time before it shows the change.

We find similar physical delay in two of other three sub models which influences the final detection result. In offline model, these influences are minimized because of larger primary time unit because in offline model only the mid point value is used in a whole chunk, and mid point value can be hardly affected by the noise and misplacement happened near segmentation. However, as is discussed above, we choose smaller time unit considering the unit unification problem, so we have to figure out another method to minimize the influence of physical delay.

By further studying the design of SWaT, we can get the main reason physical delay happens in each sub model. And according to these delay event, we set threshold of probability for each sub-model to replace 0 which we use in strategy first. For example, in sub-model 1, we set 0.02 as threshold. It means that an alarm is raised if the corresponding entry in the table is lower than 0.02. In this way, we avoid a large number of mistakes raising an alarm to physical delay.

The value choice of threshold is worth discussing about. A low threshold can not efficiently reduce false positive, while the model with a high threshold may fail to detect many attack scenarios.Therefore, all the values of thresholds we decide is from several attempts. As the threshold is calculated from statistics, the update of CPD and threshold value is also quite important in real-time operation.

Table4.3 shows the evaluation of sub-model 1. we compare the sub-model we implement with offline model. As we can see in table, our model have quite close performance to chunk-based model. However, our model actually miss one attack scenarios, scenario 16.

We do some extra study on that scenario.

Table 4.3: Points evaluation of Reai-time BN sub-model 1

Model	Offline Chunk-based BN	Real-time point-based BN
Scenario 1	TP: 46; Recall: 0.0489	TP: 46; Recall: 0.0489
Scenario 16	TP: 60; Recall: 0.0832	TP: 0; Recall: 0.0000
Scenario 25	TP: 70; Recall: 0.0598	TP: 70; Recall: 0.0598
Scenario 28	TP: 395; Recall: 0.8896	TP: 395; Recall: 0.8896
Scenario 31	TP: 480; Recall: 0.8889	TP: 480; Recall: 0.8889

The detailed description of attack scenario is "Keep MV101 on continuously; Set value of LIT101 as 700mm". The timestamp of this attack is (117000,117720). We can translate the attack procedure into symbolic events: First, MV101 = FIT101 = P101 = 2(closed); LIT101 = 4(QU); Then MV101 = FIT101 = P101 = 2(closed); LIT101 = 2(SC). In offline model, these two steps have probability 0% and 4.44% in CPD, so this attack is detected. However, in our model they have 2.12% and 4.45%, which are both greater than threshold 0.02. So the model consider this attack as normal behaviour and accept it.

In conclusion, the model we implement for real-time operation test has a performance close to offline model and meet our requirement in this study.

4.4 Combined Strategy

After we get two models respectively based on TA and BN, we need to decide the strategy combining these two models. As is mentioned above, two models have their own problems. Although TA model detects many scenarios and get quite high recall, it arouses false positive explosion. BN is much less sensitive compared with TA. However, due to different kinds of noise, it can gives out some surprising alarms which are totally unreasonable.

After several attempts, we choose "AND" strategy to combine these two model. There are two main reasons:

- When using "AND" strategy to combine two models differing greatly in sensitivity, the combined model will reserve the lower sensitivity. In this way, we can solve the false positive explosion aroused by TA model.
- As TA study the sequential behaviour of time series, it can give out the suspect behaviour in some specific ranges. With these ranges, we can easily filter the single alarms given out by BN model because of noise in original alarm.

As we mention in first section of this chapter, the combined model of TA and BN still need to be combined with OOA model. And the final model still need some overall evaluations.

4.5 Evaluation

Finally, we combine our model and compare it with TABOR. The result is shown in Table4.4.

Table 4.4: Detection Result Evaluation for Real-time Model

Model	Precision	Recall	F measure
DNN	0.98295	0.67847	0.80281
SVM	0.92500	0.69901	0.79628
TABOR	0.86171	0.78803	0.82322
Real-time	0.85471	0.77321	0.81192

As we can see, the real-time model we purpose has slightly less precision and recall than TABOR model. Therefore, F-measure of real-time model is also slightly less. However, F-measure of our model is still over DNN and SVM model.

it is worth pointing that the problems happen respectively in TA and BN do not show a great influence on final result due to "AND" strategy we use in combination. In conclusion, the model we propose for real-time operation testing meet our requirement. And during this study, we also discuss about much valuable experience about real-time modelling and testing.

Chapter 5

2-TBN System

From the last chapter, we have presented the real-time intrusion detection system inspired by offline graphical mode-based approach. In this chapter, we will present another approach which can represent the combined strategy we used in last chapter and try to answer research question 3. This includes three parts: First, we introduce the basic goal and idea of this new design. Then we present the methodology we use in this new approach. Finally, we evaluate the result of detection and compare it with the performance of combined strategy which we introduce in last chapter.

5.1 Motivation

In last chapter, we present a real-time version intrusion detection system. This real-time system is inspired by Lin's graphical mode-based approach. The core of system strategy is the combination of Time Automaton and Bayesian Network.

According to performance evaluation, the detection result of this real-time system is not quite the same as the result of offline version which is presented in Lin's research. However, the real-time system still successfully detects 23 attack scenarios out of all 36 scenarios, which is much better than the performance of systems based on SVM and DNN. And we have also discussed about the reason why there is difference between the performance of our real-time model and Lin's offline model.

It is worth pointing out that the real-time system we present in last chapter is based on combined strategy which includes multi techniques. Among all these techniques, Out of Alphabet costs little and shows a high priority. However, both Time Automaton and Bayesian Network need independent model training and testing procedure, which arouses much cost of time and computing power.

In our research, we use 60s as the smallest time unit, which means the frequency of alarms is once per minute. In this case, the testing speed of this combined strategy can meet the requirement of real-time operation. However, if we want to increase the frequency, multi-

model testing can arouse big problems and makes real-time operation hard to be realized.

Driven by the desire to reduce the cost of model, we want to find a single technique which can replace the combined strategy of multi techniques. And after several attempts, we choose Two Time Slice Bayesian Network(2-TBN) and propose a new methodology based on this technique. In this way, we try to answer the research question 3 mentioned above.

5.2 2-TBN

We have introduced Bayesian Network in last chapter. The Bayesian Network we use in multi-technique model is a time-static one, which means the probability distribution in that Bayesian network doesn't change with time. In contrast, there is another type of Bayesian Network, Dynamic Bayesian Network.

Dynamic Bayesian Network(DBN) are static Bayesian networks that are modeled over an arrangement of time-series[11, 30]. In a Dynamic Bayesian Network, variables are related to each other over adjacent time steps.

Figure5.1 shows an example of Dynamic Bayesian Network. The left part of the figure shows a simple time-static Bayesian Network. The event based Bayesian Network is only decided by state in same timestamp. The right part of figure shows Dynamic Bayesian Network which tranforms from the time-static network. In Dynamic Bayesian Network, event is decided by states in both same and previous timestamp. In this figure, Dynamic Bayesian Network is two time slice type, which means at any point in time T, the value of event can be calculated from the internal regressors and the immediate prior value(time T-1). Also, we can extend Two Time Slice Bayesian Network(2-TBN) to N time Slice Bayesian Network(N-TBN). In N-TBN, the value of event can be decided by internal regressors and N-1 prior values.

We develop our methodology based on Dynamic Bayesian Network and try to use single model to replace multi-model method in last chapter.

5.3 Methodology

Before we start developing methodology, we need to review the basic ideas in multi-model method we present in last chapter.

Besides Out of Alphabet(OOA), we use Time Automaton and Bayesian Network as basic techniques. Time Automaton(TA) detects anomalies in two ways. The first one is that TA gives out alarms when there are invalid events at given state. The other one is that TA gives out alarms when event duration is outside valid timing guard. Bayesian Network(BN) gives out alarms when there is a zero probability entry in conditional probability distribution.

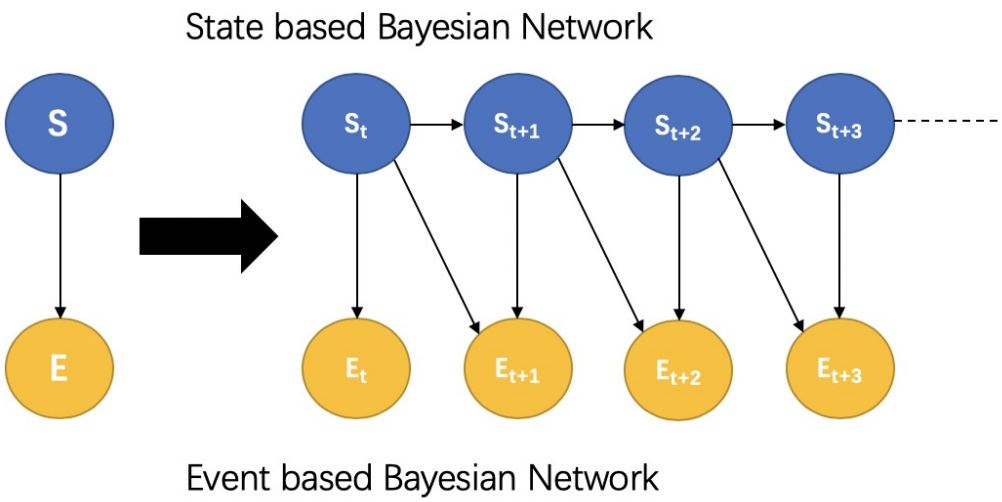

Figure 5.1: Daynamic Bayesian Network

Together with OOA, we can see that the basic idea behind all three techniques is the same. In this combined strategy, all we want to do is to find zero probability events and give out alarms to them. These events includes both concurrent events and transition events.

In multi-model method, TA helps finding transition events and BN helps finding concurrent events. We use ”AND” strategy to prove the results from two techniques and then get the well performance. To replace the combined strategy with single technique, we must fxind out both concurrent events and transition events in single model.

5.3.1 2-TBN Modelling

Dynamic Bayesian Network(DBN) includes both internal regressor relationship and time slice relationship, which seems to meet the requirement in this case. Inspired by this idea, we start the research.

We still use LIT101 as example. In Figure4.9 we have presented time-static Bayesian Network learned from training dataset. We keep the directed acyclic graph(DAG) structure and can easily get the Dynamic Bayesian Network structure, which is shown in Figure5.2. We also use a greedy search algorithm K2[10] to prove the structure shown in the figure.

The structure we use here is 2-TBN type. We also try different values for N in N-TBN type. However, even value 3 will arouse state-explosion problem. And the cost of computing power and time, which does not serve our purpose. That is why we use 2-TBN as our DBN type.

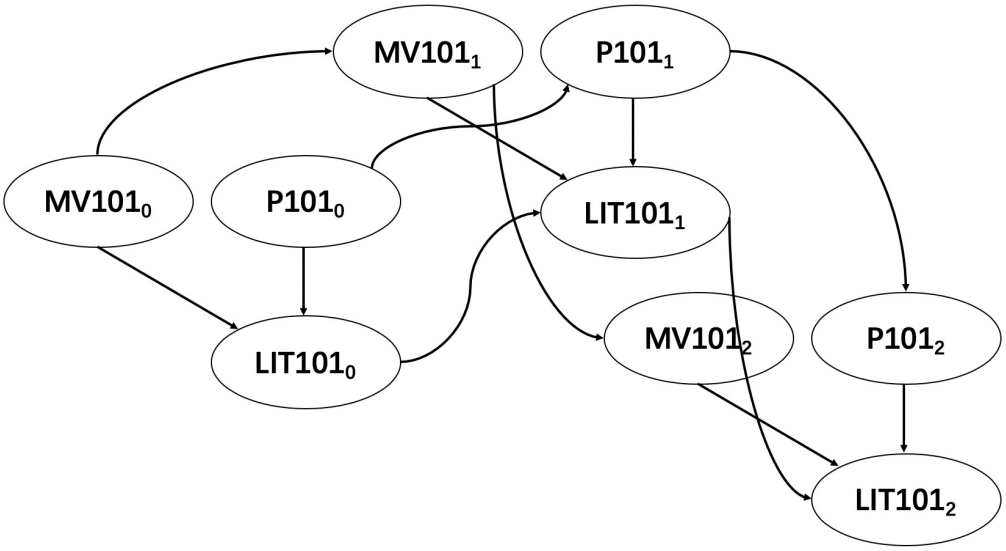

Figure 5.2: Daynamic Bayesian Network learned from LIT101

After we get the network structure of 2-TBN, we can easily calculate the conditional probability distribution of LIT101. Some examples of CPD is shown in Table5.1.

Table 5.1: CPD learned from LIT101

MV101'/P101'/LIT101	LIT101'=1	LIT101'=2	LIT101'=3	LIT101'=4
1 1 1	0	0.032	0	0
1 1 2	0	0.771	0	0
1 1 3	0	0	0	0
1 1 4	0	0.197	0	0
2 1 1	0	0	0.003	0.012
2 1 2	0	0	0	0
...	...	...	...	...

As the CPD is quite long, only some examples are shown in table. The content of this table need to be explained. The value of LIT101 means the prior value of sensor LIT101, and the value of LIT101' means the value of the sensor at this time point. Values of MV101' and P101' mean values of actuators at this time point. Among these values, we use 1 and 2 to show two different states of actuator. And also, we use 1, 2, 3 and 4 to show four different states we decide in sensor signal preprocessing. 1, 2, 3, 4 are symbols of Quick Down, Staying Constant, Quick Up, Slow Up. The numbers in table show the probability of every condition. In same way, we can get the Dynamic Bayesian Network CPD of DPIT301,

LIT301 and LIT401 as well.

5.3.2 CPD study

As we mention above, the basic idea is to find out zero probability events. Now we have the model built on 4 sensors simply based on 2-TBN, next step is to study the probability distribution in this model.

We start with the simple idea first, which is using the CPD directly. As we have got CPD of sensors, we check the CPD and give out alarms to every zero probability event shown in CPD tables. This method is easy to implement. However, the performance is quite bad. The biggest problem is about false positive points. Although we detected 47124 attack data points out of all 53866 attack data points, we also get over 200 thousand false positive points, which is even over half of all data points. Obviously, the result is not acceptable.

According to previous experiments, we can find that the same false positive explosion happens in TA-only model evaluation. In multi-technique model, we use AND strategy to prove the results of TA-only model and BN-only model each other. In that way, we solve false positive explosion problem. In CPD of 2-TBN, both transition events and concurrent events are included. So the basic idea is to make two types of events prove each other.

By studying the CPD deeply, we find that zero probability events in this CPD actually shows the OR strategy result of transition events and concurrent events.

For example, according to dataset study, when MV101'=1, P101'=1, the only valid value of LIT101' is 2. It can be confirmed by the CPD table. As we can see in first four rows, regardless of the value of LIT101, the value of LIT101' can not be 1, 3 or 4.

On the other hand, according to dataset study, when LIT101=2, the valid values of LIT101' is 1 and 2. So regardless of the values of MV101' and P101', the value of LIT101' can not be 3 or 4 when LIT101=2.

To get the separate information of transition event and concurrent event, we can merge the CPD table in two ways. Table5.2 and table5.3 show the result of merging.

Table 5.2: Merged CPD: Concurrent Event

MV101'/P101'	LIT101'=1	LIT101'=2	LIT101'=3	LIT101'=4
1 1	0	1	0	0
2 1	0	0	0.052	0.948
1 2	0.948	0.052	0	0
2 2	0	0.044	0.956	0

Table 5.3: Merged CPD: Transition Event

LIT101	LIT101'=1	LIT101'=2	LIT101'=3	LIT101'=4
1	0.768	0.042	0.118	0.072
2	0.204	0.796	0	0
3	0	0	0.813	0.187
4	0.028	0.162	0.069	0.741

As we can see, the merged CPD which shows concurrent events is just the CPD we get from time-static Bayesian Network in last chapter.

We can conclude zero probability event from both tables:

- **Concurrent Events**:
  LIT101'(MV101'/P101') = 1(11, 21, 22); 2(21); 3(11, 12); 4(11, 12, 22)
- **Transition Events**:
  LIT101'(LIT101) = 1(3); 2(3); 3(2); 4(2)

We can use two examples to explain these zero probability event. In concurrent event, 1(11, 21, 22) means that the probability of "LIT101'=1" is zero when the values of MV101' and P101' are among (1,1), (2,1) and (2,2). On the other hand, in transition event, 1(3) means that the probability of "LIT101'=1" is zero when the value of LIT101 is 3.

5.3.3 Combined Zero Probability Events

The next problem is how to use this separate information to solve false positive explosion. Our first idea is to use AND strategy directly on these zero probability events. The combined zero probability is shown as:

- **Zero Probability Events**:
  LIT101'(MV101'/P101'/LIT101) = 1(113, 213, 223); 2(213); 3(112, 122); 4(112, 122, 222)

Then we model this list and evaluate the performance. We get several conclusions:

- This model based on zero probability events detects 11 different attack scenarios among all 36 scenarios. Compared with Lin's multi-technique model, only 1 scenario is missed.
- The false positive explosion has been solved with this model. The false positive points drops from over 200 thousand to around 2 thousand.

- Compared with Lin's offline model and real-time model presented in last chapter, the recall of this model is quite low. It can be observed that only very few data points have been detected in several long-term attack scenarios.

The first two conclusions shows that concurrent events and transition events have successfully proved each other. However, the third conclusion shows that more improvement is needed.

The main reason why recall drops so rapidly is that this model is not good at locating the end of an attack. According to the study of detection result, this model can correctly locate the start of many attack scenarios. However, once an attack starts, it can not judge whether the attack ends. So in many attack scenarios, only some data points near the start of attack have been detected.

5.3.4 Final Methodology

To solve this problem, we are inspired by the idea behind the combination of TA and BN models. In real-time system presented in last chapter, we use TA model to get the approximate range and then use BN model to decide the final result. Based on this idea, we get our final methodology:

- First, we process the testing data into the form which we introduced in chapter 3. Here we use a series of data points as example
  3 1620 2 660 1 660 3 840 4 600 2 780 3 2760 ...
- We set 60s as smallest time unit. For each time unit we receive in real-time operation, we check it in transition zero probability events list:
  LIT101'(LIT101) = 1(3); 2(3); 3(2); 4(2)
  In this case, anomaly will be detected when we receive the first unit of event 2 because of the zero probability event 2(3), and this unit is set as the start of alarm:
  3 1620 2 60 ...
- Once we find the start, we set the second normal state(in this case state 3) after the start as the end of this attack(Reason has been explained in last chapter.).
- Finally, we check the follow data units before the set end in concurrent zero probability events list and get final result:
  LIT101'(MV101'/P101') = 1(11, 21, 22); 2(21); 3(11, 12); 4(11, 12, 22)

As we can see, although concurrent and transition zero probability events lists are checked separately, only a single 2-TBN model need to be trained, which reduce much cost of computing power and time compared with multi-technique model used in Lin's research.

The flow chart of methodology proposed is shown in Figure5.3.

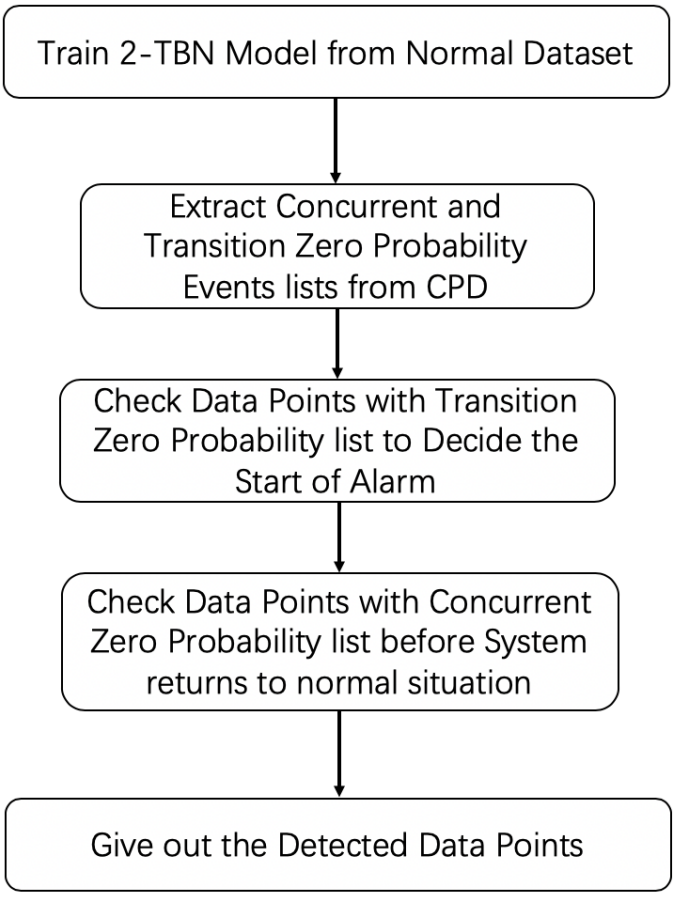

Figure 5.3: Flow Chart of Methodology

5.4 Evaluation

Finally we do some evaluations on this model's performance.

First, we check the detection result of 2-TBN model and compared it with Lin's offline graphical method and other machine learning method. It is shown as Table5.4.
It is worth pointing out that the results do not include 12 attack scenarios detected by OOA technique.

As the basic idea behind TA&BN model and 2-TBN model is the same, the result is almost the same. The most obvious difference exists in the result about attack scenario 16.

The detailed description of attack scenario 16 is keeping MV-101 on continuously and setting the value of LIT101 as 700mm. The timestamp of this attack is from 117000 to 117720. In TA&BN model, TA detects this attack because the event duration of state "Staying Con-

Table 5.4: Points evaluation in each scenario

Model	DNN	SVM	TA&BN	2-TBN
Scenario 1	0	0	0.049	0.049
Scenario 2	0	0	0.930	0.930
Scenario 7	0. 927	0.919	0.612	0.612
Scenario 13	0	0	0.597	0.597
Scenario 16	0	0.0167	0.083	0.000
Scenario 21	0	0	0.999	0.999
Scenario 22	0	0	0.196	0.196
Scenario 25	0	0.003	0.999	0.999
Scenario 28	0	0	0.890	0.890
Scenario 29	0	0	0.990	0.990
Scenario 30	0	0	0.258	0.258
Scenario 31	0	0.119	0.889	0.889

stant" is outside valid timing guard. However, although 2-TBN model can detect all the event error TA&BN model detects, it can do nothing with timing error, which makes attack scenario 16 missed in detection. This is a shortcoming compared with multi-technique model.

According to our purpose mention in section 5.1, another important aspect need to be evaluated is cost of computing power and time.

Table5.5 shows the runtime comparison. In training part, TA&BN model and 2-TBN model are highly efficient compared with SVM and DNN. There are two main reasons. First, in TA&BN and 2-TBN model training, training data is partitioned into groups and segmented, which dramatically reduce the dimension and size of training data. The other reason is that the computation complexity of learning TA, BN and 2-TBN are all polynomial time. Compared with TA&BN model, 2-TBN model still has advantage in training cost as 2-TBN model only includes single DBN model.

In testing part, we use the testing time per smallest time unit to do the comparison. The literature of SVM and DNN do not give out the detailed information about smallest time unit. So we regard that they use sampling frequency as testing frequency. In this case, SVM is the most efficient in testing among all four methods. However, the computation complexity of testing in TA, BN, 2-TBN and SVM are all O(1). So the difference among all four methods is not obvious here. 2-TBN model seems to have a little advantage over TA&BN model.
The further overall comparison is shown in Table5.6.
According to this further comparison, 2-TBN model has slightly less precision and recall than TABOR model, which makes F-measure is also slightly less than TABOR model.

Table 5.5: Runtime comparison

Model	Training	Testing
DNN	2 weeks	0.064s
SVM	30 min	0.013s
TA&BN	214s	0.051s
2-TBN	168s	0.031s

Table 5.6: Further Comparison

Model	Precision	Recall	F measure
DNN	0.98295	0.67847	0.80281
SVM	0.92500	0.69901	0.79628
TABOR	0.86171	0.78803	0.82322
Real-time	0.85471	0.77321	0.81192
2-TBN	0.85563	0.76821	0.80957

There is no obvious difference between the performance of 2-TBN model and TABOR model. And also, 2-TBN model has a F-measure over DNN and SVM model.

In conclusion, we propose 2-TBN model in this chapter to replace the multi-technique model TABOR which is combined with TA and BN. The proposed model is based on two time slice Dynamic Bayesian Network. It can detect most attack scenarios which can be detected by TABOR, but it can not detect the timing error which can be detected by TA model. The final detection result is quite close to the result of TABOR. And 2-TBN model has slightly advantage over TABOR in cost of computing power and time. It is confirmed that 2-TBN model proposed can meet our requirement in this case.

Chapter 6

Discussions

In this chapter, we summarize our work from several perspectives. We first present a summary of this work and then explain the deficiencies of this research. In the conclusion section, we answer the research questions we put forward in chapter 1. Finally, we suggest several future work perspectives.

6.1 Summary

The main contribution of this work is shown mainly in two parts.

First, we confirm that TABOR, the offline graphical model-based approach for anomaly detection in industrial control system can be implemented into real-time version and meets the requirement in real-time operation. In real world anomaly detection, real-time operation is a basic requirement for detection methods. TABOR proposed by Lin gives out a bright prospect of graphical model-based approach. However, there are many difference between offline model and real-time version. We do a lot of work in both data preprocessing and modeling procedure to increase the robustness of the whole detection system and make the system respond to incoming data more quickly in real-time operation.

In data preprocessing, we improve the procedure of denoising and alignment to deal with the problem that we can not get stable chunk segmentation in real-time operation and enable system to respond more quickly. In modeling, we separate the combined strategy used in TABOR, turn each part into real-time version and then combine them again. In this way, we realize the real-time graphical model for ICS problems. We discuss about different strategy we use in TA timing error and unit unification in BN model training. Compared with offline version, real-time model can not give out timing error without delay. And smaller time unit used in BN modelling arouses much noise caused by physical delay, which also has influences on the system performance. These study can bring other researchers a lot of experience in the field of real-time model design.

When it comes to the final result, we find that the real-time system based on combined tech-

nique strategy we implement has a almost close performance compared with offline TABOR model, which can still meet the requirement. We can conclude that this part of study in our thesis is successful and meaningful.

The other contribution of our study is that we propose 2-TBN model to replace the combined technique strategy, TABOR. The motivation of this study is to simplify the strategy and use a single model to detect both concurrent events and transition events. We use two time slice Dynamic Bayesian Network as basic technique, and model our system according to zero probability event list extracted from CPD in 2-TBN.

We record the whole study procedure in this part, including several unsuccessful attempts. These attempts can help other researchers understand 2-TBN more deeply and find out more use of it. The final methodology we propose gives out almost close performance compared with offline TABOR model. With slight cost of performance, it greatly simplifies the procedure of model training and get a slight advantage over TABOR in the cost of computing power and time.

To summarize, our study give out two real-time model used in real-time intrusion detection in industrial control systems. The first one is inspired by TABOR model, which is proposed by Lin. We implement Lin's combine technique strategy into real-time version. Besides the implementation, we take some action to increase the robustness and reaction speed of system in real-time operation and draw some valuable conclusions. The other real-time model we propose is based on 2-TBN technique. We use this single model technique to simplify the strategy used in TABOR and get the close performance successfully. Both of our methods keep graphical model-based approach's advantage in anomaly location and explanation. The 2-TBN even realize the advantage in modelling and time price with slight cost of performance. Our study confirms the bright prospect of graphical model-based approach in cyber physical system.

6.2 Answering Research Questions

From the outcomes of this research, we answer the research questions in the first chapter:

RQ1: What action should we take in data preprocessing to help increase the robustness and respond speed of detection model in real-time operation?

We take some important action in data preprocessing procedure. On one hand, we find that denoising directly decides system's reaction speed from the beginning, so the sampling frequency of denoised data worth deep study before modelling. We check the final result of data segmentation and find that in part of sub-models, we can average at least 30 origin data samples as one denoised data point. Compared with Lin's 60s per unit, we double the frequency of system reaction in some part of proposed real-time system, which means our

model respond to incoming data more quickly. And we recommend that this action should be taken again when different dataset is used. Then when the quality of dataset increases, less origin data samples can be averaged into every denoised unit and the reaction speed will increases correspondingly.

On the other hand, we find that one big resource of artificial noise is alignment of sensor and actuator data. The methodology of alignment have a big influence on the final result of data representation, which decides the performance of event-based model to a large extent. We do some experiments and find that mid-point-value method proposed by Lin arouses much noise when denoised sampling frequency increases. Compared with it, we recommend majority vote in alignment of sensor and actuator data. Detection model using majority vote arouses much less noise even when reaction speed increases gradually. In this way, the robustness of model is guaranteed.

RQ2: Can we solve the real-time intrusion problem with the combined strategy of TA, BN and OOA? What is the difference between offline TABOR model and this real-time model? How does it perform?

In this study, we implement these different techniques into real-time version separately and then combine them again. In our study, we find that there are some obvious differences between offline model and real-time model, and we take some reasonable action or fix to guarantee the robustness and detection performance of real-time model we propose. In TA part, we use a different strategy to deal with timing error. Our model only labels the over-limit part of events as attack other than the whole event. We believe it is more suitable in real-time operation. The result of detection also confirms this conclusion. Our strategy arouses fewer false positive alarms compared with TABOR's TA submodel, which improve the detection performance.

In BN part, Lin uses chunk-based modelling as all the chunks have been already divided in offline training and testing. However, in real-time operation the incoming data is dropped into system continuously and system needs to respond to incomplete chunks. In this case, unit unification between training and testing will be broke if we use chunk-based training model for testing. On the other hand, if we set the system to respond when every chunk ends, then the reaction speed will be too slow to meet the requirement of real-time operation. Therefore we use point-based model in BN modelling part to guarantee the robustness of system. And to deal with the noise aroused in this action, we set several thresholds in each sub-model to replace 0 probability as evaluation index of attack.

Finally, we evaluate the performance of final real-time model we propose. We find that the real-time system based on combined technique strategy we implement has a almost close performance compared with offline TABOR model, which can meet the expected requirement of both correct rate and real-time operation. So we can conclude that we solve the real-time intrusion problem with the combined strategy of TA, BN and OOA in this research.

RQ3: Can we find any method to simplify the model and increase testing speed compared with combined strategy? Can the model newly proposed meet the requirements of both correct rate and testing speed? How does it perform?

We study the computation complexity of testing in TABOR and our proposed model in Chapter 4 and find that the computation complexity of both TA and BN model is just O(1). So we focus on single model approach to reduce the time cost. Finally, we propose the 2-TBN model which is based on two time slice Dynamic Bayesian Network. The 2-TBN model use a single model to detect both concurrent and transition error, which realizes the goal to simplify the model. According to the result of experiment, 2-TBN model has an obvious advantage over TABOR and real-time model with combined strategy, while 2-TBN's performance of detection is almost close to these multi-model approach. So we conclude that 2-TBN model we propose in this research meets the requirements of both testing speed and correct rate. The advantage of time cost is not at a great price of performance.

With all the conclusion we get from three research question, we can say that we can surly solve intrusion detection problem in real-time operation with graphical mode-based approach. The real-time model we propose in Chapter 4 is based on TABOR. It maintains TABOR's advantage in interpretation and localization of anomalies and also meets the requirement of real-time operation. The 2-TBN model we propose in Chapter 5 uses a single model and realize the goal of simplifying. It has an advantage in time cost while the price of performance is quite slight. Both models can confirm that we complete our goal in this research.

6.3 Further Work

There are several deficiencies in this research. According to these deficiencies, we have several ideas about further work to share.

6.3.1 Methodology

Although we propose a single technique model to simplify the modelling strategy, we still can not find a way to replace OOA in our study. As we mentioned in Chapter 4, OOA detects 12 attack scenarios among all 36 scenarios with very little price. It shows high priority in our study. In real world intrusion problem, OOA is also a very common method to choose. In most cases of supervised problems, we can not get great performance with OOA only. However, we can always get our models improved with the help of OOA. In our study, we claim that the proposed method is single model technique, but we still can not deny the use of OOA.

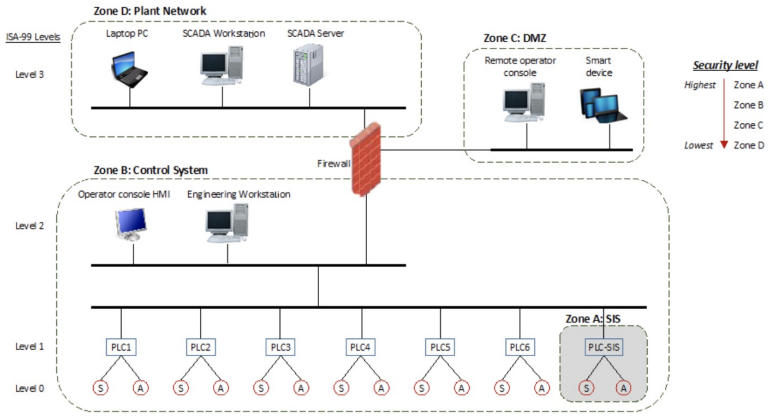

Figure 6.1: SWaT Control Network Architecture

So we suggest that in further work, there should be more work about further simplification of model strategy, which makes model easier to meet the requirement of real-time operation when sampling frequency increases.

6.3.2 Dataset

Actually the testing dataset we use in this study is not quite sufficient.

As intrusion detection problem is a real world question, the dataset we need in this study should be more realistic. Although the testing dataset we used is from real attack simulation on testbed SWaT, which includes 36 different attack scenarios, the contents in this dataset is not abundant enough for our study. First, each attack scenario happens only once in this dataset, which makes it hard to do control experiment. Secondly, each attack happens separately, which undoubtedly minimizes the difficulty in detection. Finally, as we mentioned in our study, only one of these 36 scenarios is about invalid time duration, which makes it hard to confirm the importance of state-besed model strategy.

In supervised learning, the performance of model highly depends on the quality of dataset. So we suggest that more study should be done in dataset collecting. The testbed SWaT should be made full use of in simulation and operation.

6.3.3 Network System Detection

In our study, we focus on the intrusion detection study in cyber physical system. In our study, we assume that the attacker is sitting inside the network and can operate the system directly. However, in real world intrusion, attackers often apply attacks by using network to send packages. SWaT also has a control network and I/O. Figure6.1 shows the control network architecture of SWaT. So we suggest that the behavior modeling of network traffic and

network attack detection should also be studied on testbed SWaT. And then the combination of physical and network part can be another interesting study topic.

Bibliography

[1] Sridhar Adepu and Aditya Mathur. An investigation into the response of a water treatment system to cyber attacks. In *2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE)*, pages 141–148. IEEE, 2016.

[2] Sridhar Adepu and Aditya Mathur. Using process invariants to detect cyber attacks on a water treatment system. In *IFIP International Conference on ICT Systems Security and Privacy Protection*, pages 91–104. Springer, 2016.

[3] Sridhar Adepu, Gyanendra Mishra, and Aditya Mathur. Access control in water distribution networks: A case study. In *Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on*, pages 184–191. IEEE, 2017.

[4] Chuadhry Mujeeb Ahmed, Carlos Murguia, and Justin Ruths. Model-based attack detection scheme for smart water distribution networks. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 101–113. ACM, 2017.

[5] Daniel Barbara, Ningning Wu, and Sushil Jajodia. Detecting novel network intrusions using bayes estimators. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–17. SIAM, 2001.

[6] Alvaro A Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366. ACM, 2011.

[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[8] Edmund M Clarke and Paolo Zuliani. Statistical model checking for cyber-physical systems. In *International Symposium on Automated Technology for Verification and Analysis*, pages 1–12. Springer, 2011.

[9] Pamela Cobb. German steel mill meltdown: Rising stakes in the internet of things, 2015. URL https://securityintelligence.com/german-steel-mill-meltdown-rising-stakes-in-the-internet-of-things. Accessed January 8, 2019.

[10] Gregory F Cooper and Edward Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine learning*, 9(4):309–347, 1992.

[11] Paul Dagum, Adam Galper, and Eric Horvitz. Dynamic network models for forecasting. In *Proceedings of the eighth international conference on uncertainty in artificial intelligence*, pages 41–48. Morgan Kaufmann Publishers Inc., 1992.

[12] MJ Desforges, PJ Jacob, and JE Cooper. Applications of probability density estimation to the detection of abnormal conditions in engineering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 212 (8):687–703, 1998.

[13] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[14] Eleazar Eskin. Anomaly detection over noisy data using learned probability distributions. 2000.

[15] Wei Gao and Thomas H Morris. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *Journal of Digital Forensics, Security and Law*, 9(1):3, 2014.

[16] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 140–145. IEEE, 2017.

[17] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[18] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H Hartel. Through the eye of the plc: semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 126–135. ACM, 2014.

[19] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

[20] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M Poskitt, and Jun Sun. Anomaly detection for a water treatment system using unsupervised machine learning. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1058–1065. IEEE, 2017.

[21] Ian Jolliffe. *Principal component analysis*. Springer, 2011.

[22] Austin Jones, Zhaodan Kong, and Calin Belta. Anomaly detection in cyber-physical systems: A formal methods approach. In *53rd IEEE Conference on Decision and Control*, pages 848–853. IEEE, 2014.

[23] Khurum Nazir Junejo and Jonathan Goh. Behaviour-based attack detection and classification in cyber physical systems using machine learning. In *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*, pages 34–43. ACM, 2016.

[24] Yakup Koç, Martijn Warnier, Piet Van Mieghem, Robert E Kooij, and Frances MT Brazier. The impact of the topology on cascading failures in a power grid model. *Physica A: Statistical Mechanics and its Applications*, 402:169–179, 2014.

[25] Yufeng Kou, Chang-Tien Lu, and Dechang Chen. Spatial weighted outlier detection. In *Proceedings of the 2006 SIAM international conference on data mining*, pages 614–618. SIAM, 2006.

[26] Qin Lin, Sridha Adepu, Sicco Verwer, and Aditya Mathur. Tabor: a graphical model-based approach for anomaly detection in industrial control systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 525–536. ACM, 2018.

[27] Robert Lipovsky. New wave of cyberattacks against ukrainian power industry, 2016. URL `https://www.welivesecurity.com/2016/01/20/new-wave-attacks-ukrainian-power-industry`. Accessed January 8, 2019.

[28] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.

[29] Alexander Maier, Asmir Vodencarevic, Oliver Niggemann, Roman Just, and Michael Jaeger. Anomaly detection in production plants using timed automata. In *8th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 363–369, 2011.

[30] V Mihajlovic and Milan Petkovic. Dynamic bayesian networks: A state of the art. *University of Twente Document Repository*, 2001.

[31] John Mulder, Moses Schwartz, Michael Berg, Jonathan Roger Van Houten, Jorge Mario, Michael Aaron King Urrea, Abraham Anthony Clements, and Joshua Jacob. Weaselboard: zero-day exploit detection for programmable logic controllers. *Sandia report SAND2013-8274, Sandia national laboratories*, 2013.

[32] Oliver Niggemann, Benno Stein, Asmir Vodencarevic, Alexander Maier, and Hans Kleine Büning. Learning behavior models for hybrid timed systems. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[33] Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003.

[34] Paul Oman and Matthew Phillips. Intrusion detection and event monitoring in scada networks. In *International Conference on Critical Infrastructure Protection*, pages 161–173. Springer, 2007.

[35] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[36] Shashi Shekhar, Chang-Tien Lu, and Pusheng 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*, pages 371–376. ACM, 2001.

[37] Xiuyao Song, Mingxi Wu, Christopher Jermaine, Sanjay Ranka, et al. Conditional anomaly detection. *IEEE Trans. Knowl. Data Eng.*, 19(5):631–645, 2007.

[38] Keith Stouffer, Joe Falco, and Karen Scarfone. Guide to industrial control systems (ics) security. *NIST special publication*, 800(82):16–16, 2011.

[39] Pei Sun, Sanjay Chawla, and Bavani Arunasalam. Mining for outliers in sequential databases. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 94–105. SIAM, 2006.

[40] Sicco Verwer, Mathijs de Weerdt, and Cees Witteveen. A likelihood-ratio test for identifying probabilistic deterministic real-time automata from positive data. In *International Colloquium on Grammatical Inference*, pages 203–216. Springer, 2010.

[41] Sicco E Verwer, Mathijs M De Weerdt, and Cees Witteveen. Identifying an automaton model for timed data. In *Benelearn 2006: Proceedings of the 15th Annual Machine Learning Conference of Belgium and the Netherlands, Ghent, Belgium, 11-12 May 2006*, 2006.

[42] Sicco Ewout Verwer. Efficient identification of timed automata: Theory and practice. 2010.

[43] Andreas S Weigend, Morgan Mangeas, and Ashok N Srivastava. Nonlinear gated experts for time series: Discovering regimes and avoiding overfitting. *International Journal of Neural Systems*, 6(04):373–399, 1995.

[44] Sharon Weinberger. Computer security: Is this the start of cyberwarfare? *Nature News*, 474(7350):142–145, 2011.

[45] Kyle Wilhoit and Seiki Hara. The real world evaluation of cyber-attacks against ics system. In *Society of Instrument and Control Engineers of Japan (SICE), 2015 54th Annual Conference of the*, pages 977–979. IEEE, 2015.

[46] Ji Zhang and Hai Wang. Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance. *Knowledge and information systems*, 10(3): 333–355, 2006.

[47] Xi Zheng and Christine Julien. Verification and validation in cyber physical systems: research challenges and a way forward. In *2015 IEEE/ACM 1st International Workshop on Software Engineering for Smart Cyber-Physical Systems*, pages 15–18. IEEE, 2015.