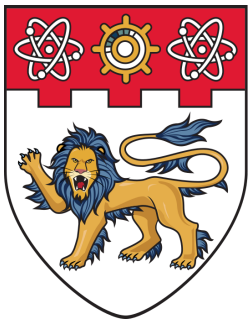


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

ANOMALY DETECTION IN MULTIVARIATE TIME SERIES USING ENSEMBLE METHOD

LIU YANLING

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2021

ANOMALY DETECTION IN MULTIVARIATE TIME SERIES USING ENSEMBLE METHOD

LIU YANLING

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirements for the degree of
Master of Engineering

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

21/09/2021

.....

Date

.....

[LIU YANLING]

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

21/09/2021

.....

Date

[A/P Chng Eng Siong]

Authorship Attribution Statement

This thesis does not contain any materials from papers published in peer-reviewed journals or from papers accepted at conferences in which I am listed as an author.

21/09/2021

.....

Date

.....

[LIU YANLING]

Acknowledgements

I would like to express my sincere gratitude and appreciation to my supervisor, Assoc Prof Chng Eng Siong, my company supervisor Dr Li Ye, and Dr Rajendra Prasad Sirigina for their guidance and supervision. I would also like to extend my thanks to the Singapore Economic Development Board (EDB) and Xylem Water Solutions Singapore Pte Ltd for the support and funding of this MEng project. I have learned valuable knowledge and explored new research areas in the project.

Contents

Abstract

Water distribution networks (WDNs) are essential services to people's life and production. The identification of anomalies and mitigation of cyber-attacks are crucial to ensure uninterrupted water service. Among various solutions of anomalies detection, matrix profile is recognized as the most time-efficient distanced-based approach. Matrix profile identifies discords in univariate time series (UTS). As the physical processes are interdependent in water networks, the data obtained from different sensors are correlated to detect anomalies in multivariate time series (MTS). However, this approach only collects positive predictions and has limitations in eliminating false-positive detections.

To improve the above-mentioned anomaly detection limitation of matrix profile, we propose and demonstrate two methods, the matrix profile with autoencoder method and the boosting method. Autoencoder, an artificial neural network trained to copy its input to its output, is introduced to reduce false alarms. Moreover, the localization of anomalies is automated by analyzing the UTS anomaly detection results. Boosting is an ensemble learning algorithm that focuses on correcting misclassified labels by the previous model with the current model. It converts weak learners to strong learners sequentially. Three boosting methods, including XGBoost, LightGBM, and CatBoost, are studied to tackle the classification of anomalies. Specifically, the proposed matrix profile with the autoencoder based ensemble model is applied as a semi-supervised anomaly detection model. The three boosting-based models are proposed as supervised anomaly detection models.

To validate effectiveness in complex environments of water distribution system (WDS), we tested the proposed two methods with simulated datasets containing labeled cyber-attacks. Both the matrix profile with autoencoder model and the CatBoost model show high accuracy of 0.9645 and 0.9245, respectively, superior to the existing state-of-the-art models. In addition, the boosting methods are also applied to anomaly detection on a simulated leakage dataset that contains detailed leakage information in WDS. The LightGMB provides outstanding classification results with 0.945 and 0.985 accuracy, which is competitive among the frontier models.

Publications

This thesis has not been published in any conference or journal.

List of Figures

List of Tables

List of Abbreviations

WDS	Water Distribution System
WDN	Water Distribution Network
MTS	Multivariate Time Series
UTS	Univariate Time Series
SCADA	Supervisory Control and Data Acquisition
IWSA	International Water Supply Association
BATADAL	the BATtle of the Attack Detection ALgorithms
LeakDB	the Leakage Diagnosis Benchmark
RSI	RetailSales Index
PAA	Piecewise Aggregate Approximation
APCA	Adaptive Piecewise Constant Approximation
SAX	Symbolic Aggregate Approximation
FT	Fourier Transform
FFT	Fast-Fourier Transform
DFT	Discrete Fourier Transform
WT	Wavelet Transform
EMD	Empirical Mode Decomposition
AM-FM	Amplitude and Frequency Modulated
IMF	Intrinsic Mode Function
GMM	Gaussian Mixture Model
EM	Expectation-Maximization
MAP	Maximum A Posteriori
GBDT	Gradient Boosting Decision Tree
PCA	Principal Component Analysis
ED	Euclidean Distance
SVM	Support Vector Machine
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
IDS	Intrusion Detection System
AE	Autoencoder
AUC	Area Under the Receiver Operating Characteristic Curve
ARIMA	Autoregressive Integrated Moving Average
PCA	Principal Component Analysis

ECG	Electrocardiogram
EEG	Electroencephalography
MNF	Minimum Night Flow
STD	Standard Deviation
CART	Classification and Regression Tree
GOSS	Gradient-based One-Side Sampling
EFB	Exclusive Feature Bundling

Chapter 1

Introduction

1.1 Motivation

The water distribution system (WDS) conveys water from water treatment plants to customers. It serves the demand for water from domestic, commercial, and industrial sectors. A WDS consists of multiple components, including pipelines, storage facilities, pumps, hydrants, house service connections, and other accessories [57]. Identification and analysis of anomalies in WDS are crucial for public welfare and safety. It is desirable that the public sector has the capability and capacity to in-situ monitor the whole pipeline network and infrastructure, to detect anomalies in the WDS and water quality, and to mitigate threats to water safety and security. Recent advances in sensors and communication technologies have significantly extended the sensing range, effectively enhanced the detection of anomalies in the water network, and consequently shortened the time needed by the public sector to respond to these anomalies [75]. However, the need to reconcile conflicts arising from merging information from sensors, due to big data, from large-scale water distribution networks (WDNs), puts forward new mathematical and computational challenges.

The WDS is vulnerable to different types of issues. A major and harmful failure is caused by cyber-attacks [1]. Identifying potential cyber-attacks and preventing unauthorized access to change or destroy the company's data are essential for maintaining the system's efficiency, reliability, and productivity. One of the first cyber-attacks happened in Queensland, Australia, in 2000. The supervisory control and data acquisition (SCADA) system operated by Maroochy Water Services was attacked, and it led to thousands of cubic meters of wastewater flowing into waterways and parks [81]. In 2016, an attack that changed the chemical levels of potable water from a water utility was described by the Verizon Risk Team [41]. Recent digitalization imposes new types of threats and dangers to the water sector. The reported cases of attacks on water infrastructures are increasing steadily [37] [38] [39]. Water infras-

tructure has become the third most targeted sector in cyber-attacks, while the first two are critical manufacturing and energy [39].

Another major challenge in the WDS is managing the leakage in pipe networks [52]. The water shortages due to droughts and increase in water consumption in recent years bring the urgency of having accurate leak detections on water pipelines [24]. The International Water Supply Association (IWSA) published that the overall state of the distribution system decides 20-30% of the total water production [36]. IWSA's survey claimed that leakage is the primary cause of unaccounted water loss. Leak brings both direct and secondary economic loss. The direct loss is mainly from the water, including raw water collection, treatment, and transportation. The secondary loss comes from repairing the damaged pipes, foundations of roads and buildings, and the economic activities affected due to the repairing work. Furthermore, leaks bring health risks. When the pressure drops inside the pipe, bacteria from the surrounding soil can infect the pipes.

Therefore, it is essential to develop MTS anomaly detection models and apply them in cyber-attacks and leakages detection in WDS.

1.2 Contributions

This research studies semi-supervised anomaly detection and supervised anomaly detection in MTS. Semi-supervised anomaly detection trains the model using datasets in normal operations, while supervised anomaly detection requires datasets containing normal operations and abnormal behaviors.

In semi-supervised detection, the research proposes an ensemble method for anomaly detection, which is based on matrix profile [89] and autoencoder [6]. The method starts with matrix profile based detection. It analyzes each sensor's signal, detects possible anomalies on UTS, and correlates anomalies detected by each sensor signal to provide MTS analytic results. This method is validated on simulated cyber-attacks datasets, called the BATtle of the Attack Detection ALgorithms (BATADAL) [83]. The BATADAL has three datasets with labels to indicate whether the attacks happen at a specific time. The matrix profile based detection has performance metrics: accuracy of 0.8175, precision of 0.4420, recall of 0.9705, and F1 score of 0.6060. Autoencoder is then incorporated into the model to eliminate false alarms. The ensemble model, matrix profile with autoencoder, achieves performance metrics: accuracy of 0.9645, precision of 0.9385, recall of 0.8325, and F1 score of 0.8800. The localization results are auto-generated in the process of UTS detections.

In supervised anomaly detection, three boosting methods, XGBoost [16], LightGBM [42], and CatBoost [26], are proposed to detect anomalies on MTS. They are validated on BATADAL

datasets as well. The CatBoost detection model outperforms the other two models. Its performance metrics are as follows: accuracy of 0.9245, precision of 0.9290, recall of 0.9245, and F1 score of 0.9155.

The anomaly detection models are tested using simulated leakage datasets, called the Leakage Diagnosis Benchmark (LeakDB) [86]. Since the leak and non-leak labels are provided in the LeakDB, the three boosting-based anomaly detection models are selected and utilized. One thousand scenarios in the Hanoi water network and one thousand scenarios in the Net1 water network from LeakDB are classified by XGBoost, LightGBM, and CatBoost models. LightGBM gets slightly better performance metrics among the three models. It achieves an accuracy of 0.945, precision of 0.950, recall of 0.945, and F1 score of 0.946 in the Hanoi network; an accuracy of 0.985, precision of 0.986, recall of 0.985, and F1 score of 0.985 in the Net1 network.

1.3 Organization of Report

Chapter 1 introduces the WDS as well as anomalies in the WDS, including cyber-attack. Chapter 2 reviews time series representation, anomaly detection technologies for time series, and the corpus, including previous work on the corpus. Chapter 3 presents anomaly detection using the matrix profile and the autoencoder ensemble method. Chapter 4 discusses three boosting anomaly detection models. Chapter 5 describes the application of anomaly detection models on LeakDB. Chapter 6 summarizes the contribution and future work for research.

Chapter 2

Literature Review

This chapter introduces the key concepts and components of time series, time series representation, and time series anomaly detection. It starts with introducing the background of time series and time series representation (Section 2.1), including classical decomposition (Section 2.1.1), symbolic representation (Section 2.1.2), frequency domain representation (Section 2.1.3), time-frequency domain representation (Section 2.1.4), and empirical mode decomposition (Section 2.1.5). This chapter then reviews anomaly detection algorithms over the years from five categories (Section 2.2), including statistical-based method (Section 2.2.1), nearest neighbor-based method (Section 2.2.2), clustering-based method (Section 2.2.3), classification-based method (Section 2.2.4), and deep learning-based method (Section 2.2.5). Finally, it compares public datasets and introduces the selected corpus for the research, namely BATADAL and LeakDB (Section 2.3).

2.1 Time Series Representation

Time series is an essential temporal data objects class. It translates observation into chronological information and can be obtained from economics, finance, medicine, and scientific applications. Large data size, high dimensionality, and continuous updating are intrinsic to time-series data. According to [18], time series can be represented as X_T, where each observation at time T is recorded. X_T can be written in the format:

$$\{X_1, X_2, X_3, ..., X_t\} \ or \ \{X_T\}, \ \ where \ T = 1, 2, ..., t.$$

Real-world signals are commonly aperiodic, noisy, and influenced by multiple sources. Extracting useful information from these signals is a fundamental goal of signal processing. However, capturing meaningful information from an environment may not be so simple in

practice. Table 2.1 summarizes the advantages and disadvantages of multiple time series representation and decomposition methods. Furthermore, in the following subsections, this research presents more details for each type of method. They are essential tools for feature extraction.

Method	Advantage	Disadvantage
Classical Decomposition	It can describe the time series from consistency. It can describe the time series from recurrence.	It is limited to the time domain. It requires the time series in a specific pattern.
Symbolic Representation	It reduces input dimension. It eliminates the effects of noise.	It distorts the time-series original shape.
Frequency Domain Representation	It can retrieve information from the frequency domain.	It requires the frequencies to be consistent over an entire signal.
Time-Frequency Domain Representation	It offers a simultaneous localization in the time and frequency domain. It can separate the fine details in a signal.	It lacks shift-invariance. It has poor directionality. It lacks phase information.
Emprical Mode Decomposition	It can estimate subtle changes in frequency. It allows the time-series data to remain in the time domain.	Its performance is highly affected by the noise on the signal.

Table 2.1: Summary of time series representation and decomposition

2.1.1 Classical Decomposition

A time series can be broken down into systematic and non-systematic components in classical decomposition [23] [91]. The systematic components can describe the time series from the perspectives of consistency and recurrence. A variety of patterns are exhibited from the systematic components in time series analysis. The non-systematic components are comprised of the rest of the elements in the series after eliminating the systematic components. Denoting Seasonality as S_t, Trend as T_t, and Remainder as R_t, a time series can be represented with components from an additive decomposition:

$$X_t = S_t + T_t + R_t \tag{2.1}$$

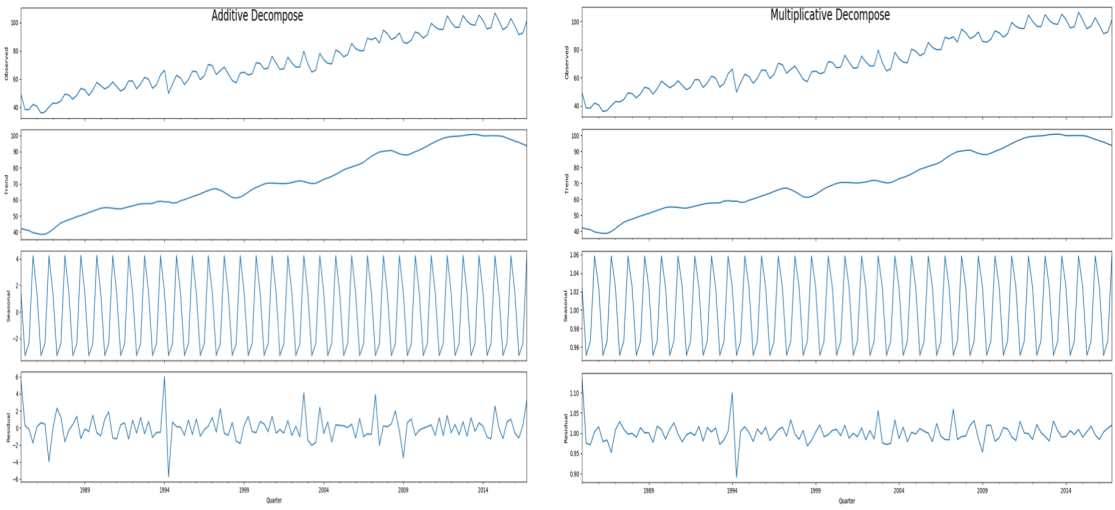


Figure 2.1: Classical decomposition of the time series using additive and multiplicative method

Additionally, a time series can also be represented with components from an multiplicative decomposition:

$$X_t = S_t \times T_t \times R_t \tag{2.2}$$

If the seasonal fluctuation or trend-cycle variation does not vary with the time series level, additive decomposition is more appropriate. The decomposition of time series can be elaborated with sales data. Retail Sales Index (RSI) [64], which uses the sales records of retail establishments to measure the short-term performance of retail industries, is introduced to elaborate the decomposition of time series with the additive and multiplicative model, refer to Figure 2.1. The increasing trend and repeated cycle are observed and obtained after the classical decomposition method is applied. In this example, multiplicative decomposition is slightly preferred as some repeated patterns are seen in the Remainder of additive decomposition.

2.1.2 Symbolic Representation

The characteristics of time series can be observed by presenting the time series with dimensions reduced from the original data set. This section introduces sampling, Piecewise Aggregate Approximation (PAA) [43], and Symbolic Aggregate Approximation (SAX) [54]. Sampling is one of the simplest methods to process the time series [8]. The dimension of the time series is reduced by a specific rate via sampling. However, sampling has a limitation as it may distort the time series shape. Figure 2.2 provides an example of how sampling works on the signal.

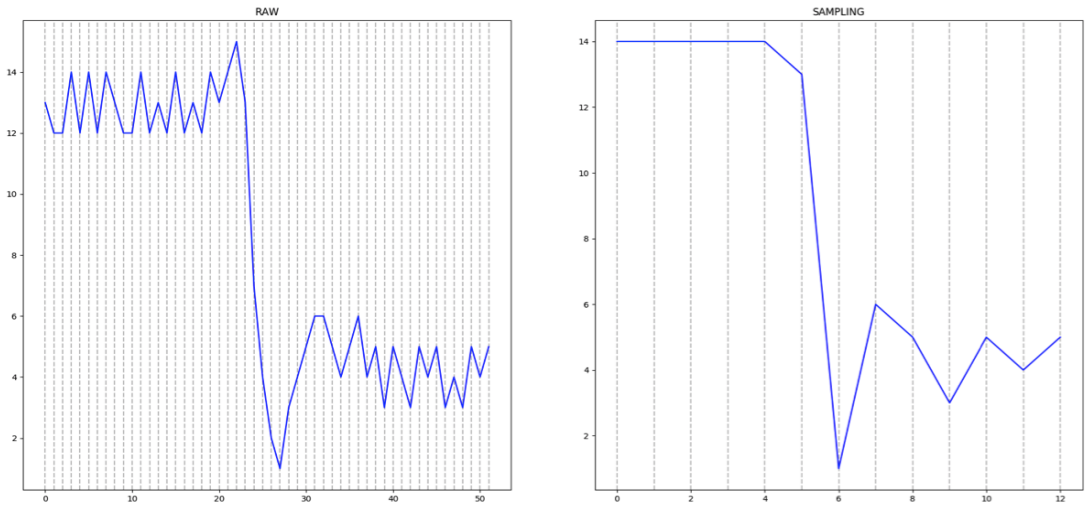


Figure 2.2: Time series dimension reduction by sampling

PAA is introduced as an enhanced method to represent time series. In PAA, the time series is divided into k segments, and the mean value of the data points in each segment is computed. These mean values form the new time series. Additionally, PAA uses segments of equal size. Figure 2.3 gives an example of applying PAA with a segment of 10 to the time series. The method is extended to adaptively analyze the segment depending on the shape instead of the fixed length, and it is called Adaptive Piecewise Constant Approximation (APCA) [43].

SAX is another widely used time series representation proposed by Lin et al. [54]. It transforms a numerical time series into continuous subsequences and then converts them into lists of symbols based on the PAA representation. The steps to achieve SAX includes:

- Divide the time series into k segments;
- Take the mean of each segment; and
- Quantize the mean values into a symbol from an alphabet of size N.

As SAX assumes that the the time series follows Gaussian distribution, the mean values are quantized based on the Gaussian distribution's breakpoints. Figure 2.4 shows an example of applying SAX with a segment of 10 and symbols of 8 to a time series.

2.1.3 Frequency Domain Representation

At the core of signal processing is the Fourier Transform (FT) [13]. The FT decomposes a function into sines and cosines, i.e., waves. In theory, any function can be represented in

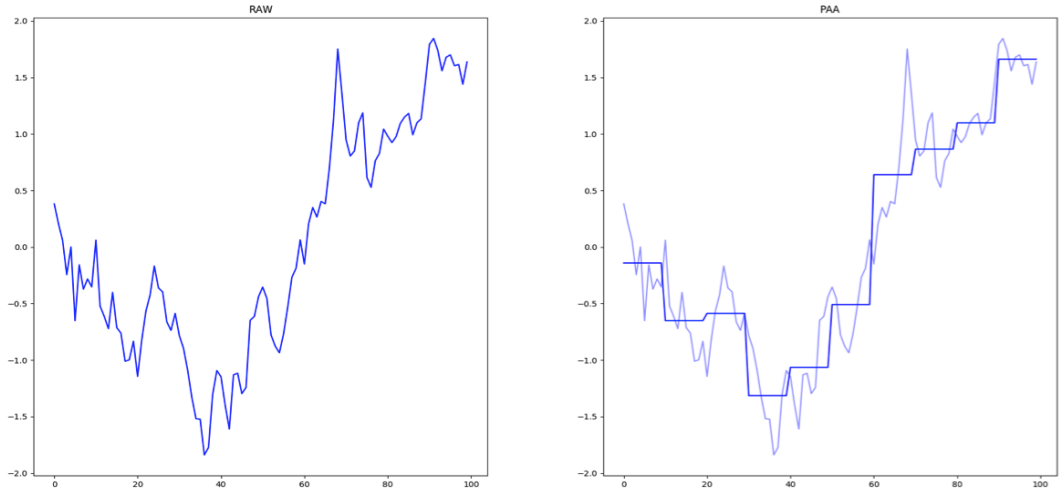


Figure 2.3: Time series dimension reduction by PAA with a segment of 10

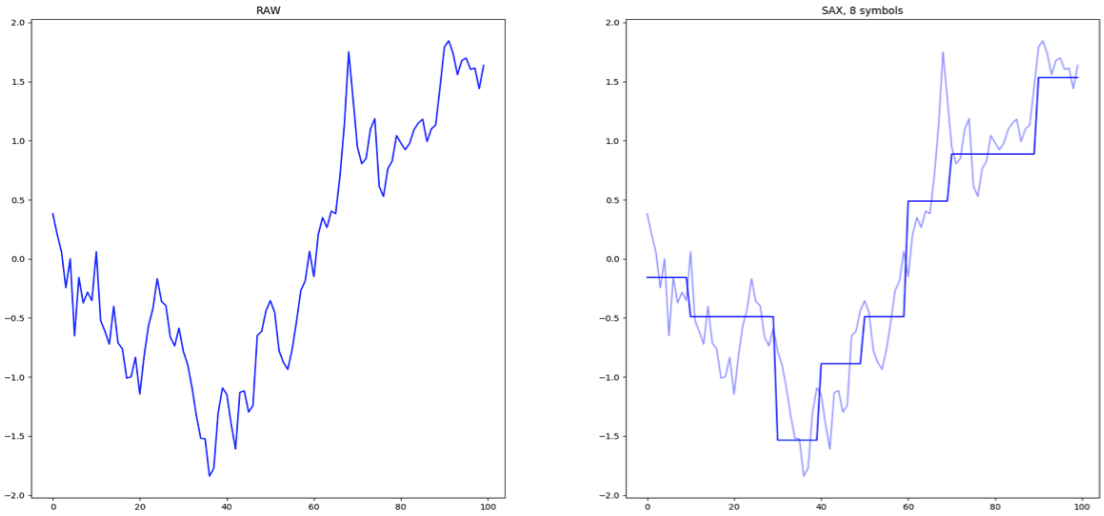


Figure 2.4: Time series dimension reduction by SAX with segment of 10 and symbols of 8

this way: as a sum of (possibly infinite) sine and cosine functions of different amplitudes and frequencies.

Fast-Fourier Transform (FFT) is developed based on Discrete Fourier Transform (DFT) [63] [32]. Garl Friedrich Gauss invented the critical factorization step in FFT around 1805. Cooley and Tukey published a joint paper in 1965 to describe FFTs [19]. FFT does not change the theory of FT but improves the efficiency of DFT. The improvement can be described in the statistic that FFT reduces the computations required for a sequence of N points from $O(N^2)$ to $O(N \log N)$. The basic step is using identity to break up the DFT of length N into two DFT of length N/2. For a given sequence $\{x_n\}$, the FFT can be written as:

$$X_n = \sum_{n=0}^{N/2-1} X_{2n} \mathrm{e}^{-2\pi i n k/N} \tag{2.3}$$

One of the significant limitations in using FT is that it requires the frequencies to be consistent over an entire signal in order to get good results [80]. It leads to the signal decomposition via FT not extracting correct features when the time series frequency changes with time.

2.1.4 Time-Frequency Domain Representation

A wavelet can be described as a wave-like oscillation localized in a time domain [92]. The wavelet transform (WT) decomposes a function into a set of wavelets. Scale and location are the two basic properties of wavelets. Scale is linked to the frequency of the wavelet. High-frequency information is captured on a smaller scale, while low-frequency information is captured on a larger scale. The property of location allows the wavelet to capture where the oscillations occur in a short interval. In the 1980s, the theory of WT had extensive development [66]. A WT decomposes the time series signal at different scales and positions. Furthermore, in a WT treatment, the mother wavelet is used to derive all basis functions, $\Psi_{a,b}(x)$, by dilation and translation as follows:

$$\Psi_{a,b}(x) = a^{-\frac{1}{2}} \Psi(\frac{x-b}{a}), \ \ where \ \ a, b \in R, a > 0 \tag{2.4}$$

In this equation, a is the scale and b is the position parameter [58] [71].

2.1.5 Emprical Mode Decomposition

N.E. Huang et al. (1998) proposed Empirical Mode Decomposition (EMD) as a non-linear technique to adaptively represent non-stationary signals in the sum of zero-mean ampli-

tude and frequency modulated (AM-FM) components [34]. The time series components partitioned by EMD are called intrinsic mode functions (IMFs). This method is utilized in non-linear and non-stationary natural signal analysis as it does not leave the time domain and provides insight into multiple signals.

An IMF is a function having only one extreme between zero crossings with a mean value of zero. Assume T is the univariate signal and k is the number of times to repeat the sifting process until an IMF is computed. The process to extract IMFs can be described with the steps as following:

1. Denote m_k as the mean of the upper and lower envelopes identified by cubic-spline interpolation. Denote h_k as the calculated component. Let T be h_0. Then h_k can be represented as:
$$h_k = h_{k-1} - m_k \tag{2.5}$$

2. Let h_k be I_1. Remains, denoted as R_1, can be computed by separating I_1 from the univariate signal T (h_0):
$$R_1 = h_0 - I_1 \tag{2.6}$$

3. Repeat the procedure n times. The residue R_n is calculated:
$$R_n = R_{n-1} - I_n \tag{2.7}$$

The signal is assumed to be stationary and linear in traditional signal processing techniques. This assumption does not work well for non-stationary vibration signals, of aging structures subjected to complex excitation, such as traffic load, high-intensity wind gusts, and earthquakes [27]. The EMD has advantages in time-series analysis. It allows the time-series data to remain in the time domain. If time-series data is transformed to another domain, like the frequency domain, the time between frequency changes is blurred if we need to retrieve precise frequency resolution. The EMD does not have this problem as its decomposed signal IMFs contain information on how the original signal changes with time.

2.2 Anomaly Detection

Before introducing anomaly detection, we briefly describe anomalies. Anomalies, also called outliers in this report, are values that deviate significantly from others, which means that the observations diverge from the overall pattern at a specific time. In other words, they represent abnormal behaviors that occur at a particular moment. Outliers can be univariate

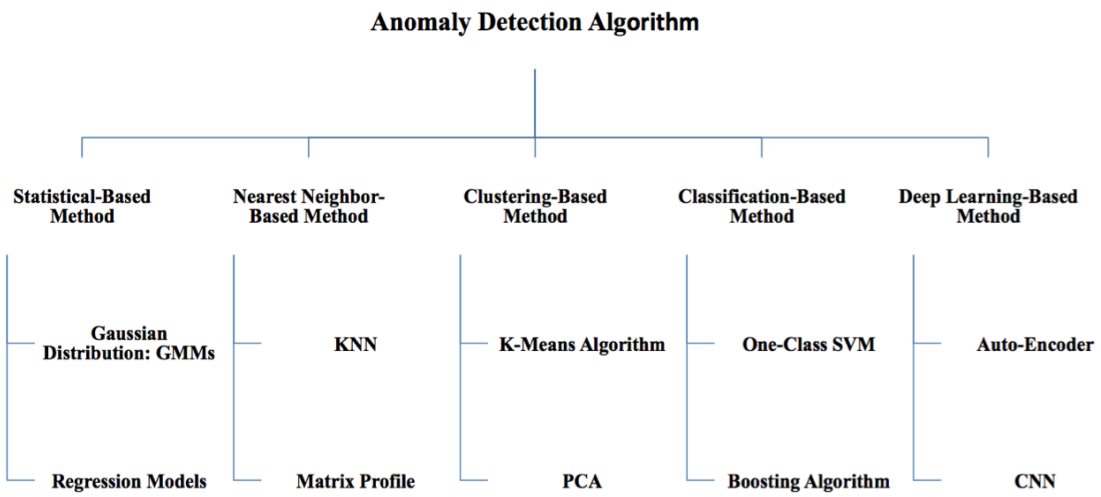


Figure 2.5: Algorithms of time series anomaly detection

or multivariate based on the time series characteristics. Additionally, they can be categorized into three types based on their content: point outliers (also called "global outliers"), contextual outliers (also called "conditional outliers"), and collective outliers [3].

Anomaly detection can be classified into three categories based on the availability of labels: supervised anomaly detection, unsupervised anomaly detection, and semi-supervised anomaly detection [45] [85]. A supervised anomaly detection model utilizes labeled data in the training process. It extracts features and studies signal behaviors when there are normal operations and abnormal events. In some scenarios, the labeled data is not available. Therefore, unsupervised anomaly detection models are implemented. An unsupervised anomaly detection algorithm clusters historical data into various groups. Then, it studies the test data to determine if the test data is far away from the current clusters. In most scenarios, the labeled data set for anomalies is not easy to collect as anomalies do not happen frequently. However, most of the time, the time series reflects the data when sensors have normal operations. In this case, semi-supervised anomaly detection models are taken into consideration. The model can be trained by learning the time series in normal states and then detecting normal operations or anomalies. In general, as shown in Figure 2.5, anomaly detection algorithms can be categorized into five types: statistical-based method [17], nearest neighbor-based method [5], clustering-based method [46], classification-based method [11], and deep learning-based method [51]. The advantages and disadvantages for each type of algorithm are summarized in Table 2.2.

2.2.1 Statistical-Based Method

The statistical-based method assumes that the time-series data follows a particular distribution [17]. It assumes that data points collected in normal behavior fall into high probability regions computed from a stochastic model, and outliers occur in low probability regions computed from a stochastic model. Anomalies can then be detected by calculating whether the data points fall into the expected probability distribution or whether the data points deviate from the standard statistical features, including mean, maximum, and minimum.

Gaussian Mixture Models

Gaussian Mixture Models (GMMs) estimate probability density by having a weighted sum of Gaussian component densities [70]. It assumes that the training dataset is in a normal distribution. The parameters of GMMs can be trained by using the iterative Expectation-Maximization (EM) algorithm or the Maximum A Posteriori (MAP) estimation.
[4] uses the probabilistic approach for time series anomaly detection on gas data. It presents a method combining distribution functions and linear regression models for detecting anomalies. The probability of test data points classified as a particular class of a prior subset is computed based on GMM distribution.

Regression model

The regression model is considered as a parametric statistical technique to achieve anomaly detection. The regression model-based anomaly detection consists of two basic steps: fit the data to the regression model in the first step, and produce the anomaly score based on the residual of each test instance in the second step [14].
The major limitation of a regression model is that if anomalies are present in training data, the regression parameters are affected, which leads to inaccurate results. To address this concern, Rousseeuw and Leroy (1987) have proposed robust regression techniques. As the anomalies usually have more significant residuals in the robust fit, the robust regression techniques can hide the anomalies and also detect the anomalies [15]. Bianco et al. (2001) and Chen et al. (2005) discussed similar robust anomaly detection algorithms integrated with Autoregressive Integrated Moving Average (ARIMA) models [12].

2.2.2 Nearest Neighbor-Based Method

The nearest neighbor analysis is widely used in anomaly detection. It assumes that data from normal behavior is located in dense neighborhoods, while anomalies are located in areas

far away from these neighborhoods [5]. The nearest neighbor-based method is generally integrated with distance-based methods and density-based methods. This technology compares the distance or similarity measured to decide whether the data points are anomalies.

K^{th} Nearest Neighbour

The basic nearest neighbor-based anomaly detection algorithm is based on the concept that a data point's distance to its K^{th} nearest neighbor in a given dataset determines whether the point is likely to be the anomaly [31]. Assume that the anomaly score is assigned to the point. Then, a threshold is generally introduced and applied to the anomaly score to decide if the tested point is classified as an anomaly.

Matrix Profile

Matrix profile is a versatile tool introduced for problems of similarity join or all-pairs-similarity search. The algorithm can be trivially cast as an anytime algorithm and produces high-quality approximate solutions in a reasonable time. Matrix profile improves performance by using the distance-based method in motifs and discords discovery [89] [20]. Chin-Chia Yeh et al. describe a case in [89]: a time series of length 525,600 is doing a similarity self-join with sub-sequences of 10,080. The nested loop algorithm requires 132,880,692,960 Euclidean distance computations and will take 153.8 days if each Euclidean distance computation takes 0.0001 seconds. Using matrix profile technology, the time taken could be reduced to 6.3 hours. Besides performance improvement, as the matrix profile is concerned with a similarity search for fixed-length subsequences, the search window length is the only input parameter required for the matrix profile computation.

2.2.3 Clustering-Based Method

Clustering-based anomaly detection techniques assume that data points can be grouped into clusters based on similarity while anomalies do not belong to any cluster [46]. The size of a cluster is set by distance. If the data point's distance to the cluster center exceeds the set threshold, it is flagged as an anomaly. Clustering is primarily thought of as an unsupervised method, though the semi-supervised clustering method was explored in [49].

K-means

K-means algorithm is a widely used clustering algorithm. It is an unsupervised learning algorithm and does not require the model trained based on a given dataset. The algorithm

predefines a number, K, and classifies the data into K classes. In the clustering process, starting from a single sample, the mean value of the clustered data is continuously calculated as the center of the entire class. Then the data closest to the center of this class is included in the same class. This method is applied to anomaly detection by defining a threshold and comparing the distance between the data point and its nearest centroid. If the distance is greater than the threshold value, then it is an anomaly [62].
[94] applies the k-means clustering anomaly detection model to detect the anomalies in the wind turbines data collected by the SCADA system. The data from the normal operating state is trained by the k-means clustering algorithm. [35] proposed a new k-means type smooth subspace clustering algorithm called TSkmeans algorithm. The proposed method can effectively exploit the inherent subspace information of a time series data set to enhance clustering performance.

Principal component analysis

Principal component analysis (PCA) is a data transformation technique. It is commonly employed for reducing data dimensions and detecting anomalies. PCA can be based on either the covariance matrix or the correlation matrix [25].
PCA was first discussed by Pearson [67] and Hotelling [33]. It was not commonly used for large datasets until electronic computers were widely available [40]. [79] describes anomaly detection using robust PCA estimating the principal components from the normal training data's covariance matrix, while [50] took the correlation matrix instead of the covariance matrix for computing the principal component scores in anomaly detection. [74] proposed anomaly detection using PCA techniques to calculate the orthogonal distance from the data point to the PCA subspace and score distance. Normal data can be identified as its orthogonal and the score distances are small.

2.2.4 Classification-Based Method

Classification is used to learn a classifier from training datasets with labeled information and then classify the test dataset into a class generated from the training process [11]. Therefore, the classification-based anomaly detection model assumes that the classifier used for distinguishing normal and anomalous classes can be learned from the given features. As shown in Figure 2.6, the classification-based anomaly detection techniques can be divided into two groups: one-class and multi-class, based on the labels from the training phase.

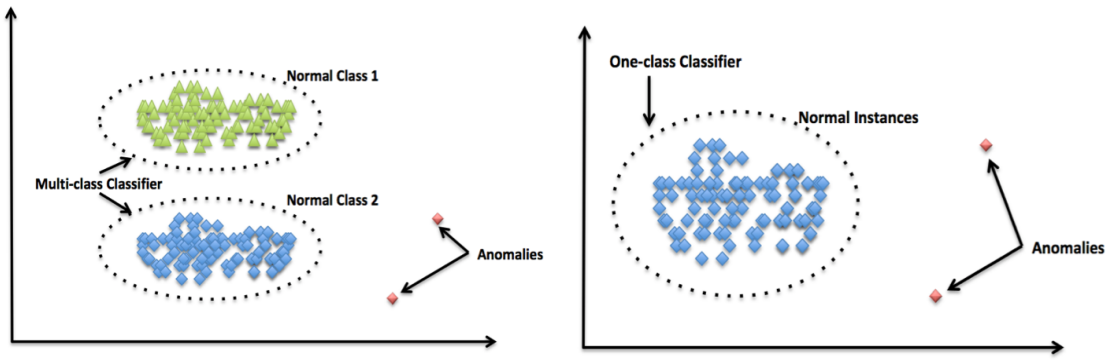


Figure 2.6: Time series anomaly detection using classification-based method

One-Class Support Vector Machine

Support Vector Machines (SVMs) have been used in a variety of classification applications. It is considered as one of the most successful machine learning techniques. SVMs use hyperplanes in multi-dimensional space to separate one class of observations from another. One-Class SVMs are designed for cases to detect anything outside the only known class. The algorithm is used for the automatic identification of unforeseen or abnormal phenomena, like outliers [53].

[93] studies the intrusion data in network security and proposes a one-class SVM model to detect the anomalies from intrusion. The model adopts the dataset in normal operations for the training stage and detects multiple attacks in the testing stage.

Boosting

Boosting refers to a family of algorithms converting weak learners to strong learners. It is supervised learning. Boosting uses the ensemble method to improve the predictions of a given learning algorithm. The boosting algorithm trains the weak learner sequentially and corrects its predecessor to get better predictions [77] [78].

Chen and Guestrin described the XGBoost algorithm in [16]. The algorithm is an approximate tree learning via sparse data and weighted quantile sketch. The paper provides insights on cache access patterns, data compression, and sharding to build a scalable tree boosting system. [22] used XGBoost technologies and achieved accuracy of 98.70% in intrusions detection.

Ke et al. proposed a novel Gradient Boosting Decision Tree (GBDT) algorithm, called LightGBM algorithm, in [42]. The algorithm uses Gradient-based One-Side Sampling and Exclusive Feature Bundling techniques to process large volumes of data and features. The paper presented that LightGBM has significant improvement in computational speed and

memory consumption comparing with XGBoost. Minastireanu and Mesnita [59] published their work to use the LightGBM algorithm for online click fraud detection. They studied the click patterns in a dataset that contains 200 million clicks in four days and obtained an accuracy of 98%.
A. V. Dorogush et al. presented a gradient boosting algorithm handling categorical features called CatBoost [26]. CatBoost deals with categorical features in the training stage instead of using preprocessing time. It uses a new schema for leaf values calculation to deduce overfitting when selecting the tree structure. J. Hancock et al. (2020) tested CatBoost on the Medicare fraud dataset in [29]. The experiments show how CatBoost eliminates some data preprocessing steps and observes that CatBoost achieves better performance than XGBoost in the Area Under the Receiver Operating Characteristic Curve (AUC) metric: CatBoost obtains an average AUC value of 0.7851, which is 0.0236 higher than the average AUC from XGBoost.

2.2.5 Deep Learning-Based Method

Deep learning has found its way into many areas and has been successfully applied to computer vision, speech recognition, and anomaly detection in recent years. Deep Learning Algorithms find associations between inputs and outputs via a neural network composed of input layers, hidden layers, and output layers. Each layer is composed of nodes [51].

Autoencoder

Autoencoder is an unsupervised artificial neural network [6]. It efficiently compresses and encodes input data, and then reconstructs the data back from the reduced encoded representation to a representation close to the original input.
V. Miranda et al. [60] trained the autoencoders for each particular fault pattern and no-fault condition to detect the faults in power transformers at the early stage. When input a new vector, the autoencoders compete with one another and recognize the closest pattern to make the diagnosis. It achieved 100% accuracy over the test dataset.

Convolutional Neural Networks and Recurrent Neural Networks

Advances in deep learning have brought a revolution in many areas of data-driven analysis, including anomaly detections. The two major network architectures are convolutional neural networks (CNN) and recurrent neural networks (RNN) [90]. CNN's are typically the first choice for image-related tasks, while RNN's are generally applied to temporal sequence tasks. For this reason, most previous deep learning work on time series anomaly detection was based

on RNN's. Malhotra et al. proposed an LSTM-based encoder-decoder scheme for anomaly detection in [55]. It learns to reconstruct "normal" time-series behavior, and after that, uses reconstruction error to detect anomalies. Kim et al. (2016) applies an LTSM-based model, and Yin et al. (2017) proposes a RNN-IDS approach to detect attacks in an intrusion detection system (IDS) and confirms the effectiveness of deep learning detection methods on IDS [44].
Some researchers are working with CNN's for time series anomaly detection. Rajpurkar et al. (2017) trained a 34-layer CNN to map the ECG samples to rhythm classes [68]. The work achieved higher performance in both recall and precision compared with six other individual cardiologists from committees of board-certified cardiologists.

2.3 Corpus

2.3.1 Summary of public datasets

Multiple public datasets are explored in this research, as indicated in Table 2.3. The selection of the dataset is based on the following conditions:

- The time-series signals contain regular patterns from normal operations and allow the model to detect contextual outliers.
- The time-series signals have events observed by multiple sensors and allow the model to detect collective outliers.
- The time-series signals contain proper labels and allow the model to have semi-supervised and supervised detection. Moreover, the dataset should preferably be from a WDS.

A quick summary of the characteristics of various benchmark datasets is presented in Table 2.3. The preferred conditions are met by datasets in BATADAL and LeakDB. Hence, in this research, we focus on datasets in BATADAL and LeakDB.

2.3.2 Summary of BATADAL

BATADAL is the collection of sensors' readings containing simulated cyber attacks in WDS [83]. It is created by faculty and researchers in iTrust, which is the world-class testbed for experimentation. These testbeds run non-stop for days with and without launching cyber-attacks. BATADAL can be used to objectively compare the performance of algorithms for the detection of cyber-attacks in WDS. The time and attack status of each dataset are described in Table 2.4. The dataset contains date-time, attack flag, and 43 variables in the

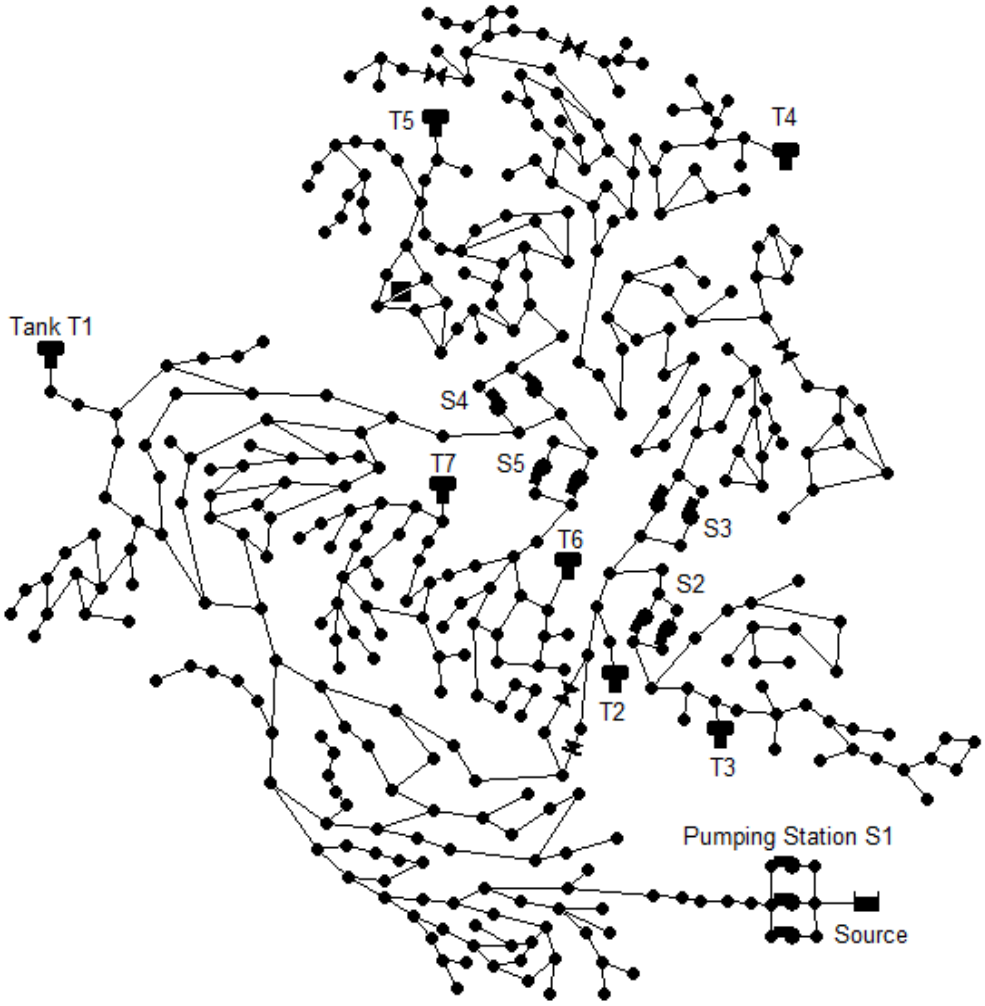


Figure 2.7: C-Town WDS

network in a selected time range. The 43 variables include seven water levels for tanks, one water flow rate for the valve, 11 water flow rate for the pumps, one state for the valve, 11 states for the pumps, and two inlet and outlet pressure for the valves distributed in the water network, C-Town. C-Town WDS is a medium-sized network first proposed by Ostfeld et al. (2012) in [65]. Figure 2.7 shows C-Town WDS loaded by EPANET tool [72]. EPANET tool is a software application to simulate hydraulic and water quality models within pressurized pipe networks. It can track pipes' water flow, nodes' pressure, tanks' water level, and a specific chemical species concentration in the simulated period. [73].

Taormina and Galelli (2018) applied autoencoder technologies to the cyber-attack time-series dataset in [82]. The proposed autoencoder (AE) model learns from the reproducible patterns of hydraulic processes, considers normal operations, and detects anomalies when

cyber-attacks occur. It benchmarks the model against XGBoost and LightGBM. The results are presented in Table 2.5. The proposed method for cyber-attack detection and localization requires good knowledge of hydraulic modeling, the operations of the C-Town SCADA system, and manual analysis of the AE errors.
Kravchik and Shabtai (2021) have studied the deep learning-based anomaly detection in this cyber-attack dataset and achieved a high level of accuracy, exceeding earlier works [48]. Their result is presented in Table 2.6. The source code is not published. Hence, we are not able to reproduce their results.

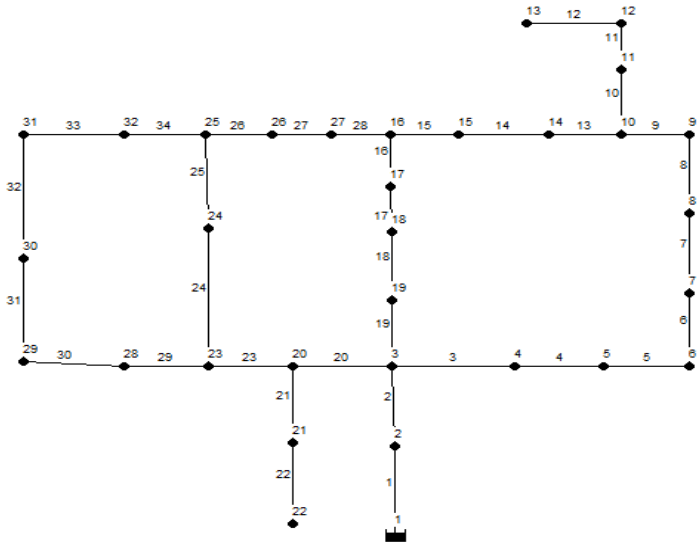
2.3.3 Summary of LeakDB

The Leakage Diagnosis Benchmark (LeakDB) is published in Jul 2018 by KIOS Research and Innovation Center of Excellence, Department of Electrical and Computer Engineering, University of Cyprus [86]. It is created by the tool of Water Network Tool for Resilience (WNTR), a python library for simulation and water networks analysis [47]. The dataset collects a large number of simulated scenarios based on realistic leakages on various water distribution networks. It targets to provide an open-source for researchers evaluating the leak detection algorithms. The LeakDB dataset contains two networks: Hanoi and Net1. Figure 2.8(a) presents the Hanoi WDS and Figure 2.8(b) presents the Net1 WDS.
Both WDS dataset obtains the leakage parameters, including the number of leaks, leak locations and leak size, the network conditions including pipe length, pipe diameter, and pipe roughness, the hydraulic model including flow, pressure, and demand in given nodes and links via files shown in Figure 2.9. The characteristics of the datasets in Hanoi and Net1 WDS are summarized in Table 2.7 and Table 2.8.
Vrachimis and et al. make use of minimum network inflows in the night time to detect the anomalies, called minimum night flow (MNF) detector. Assume MNF(k) is the minimum night flow in the k^{th} day and min{MNF(i, w)} is the minimum night flow of a moving window w days. The difference, D(k), between MNF(k) and min{MNF(i, w)} can be calculated by Equation 2.8.

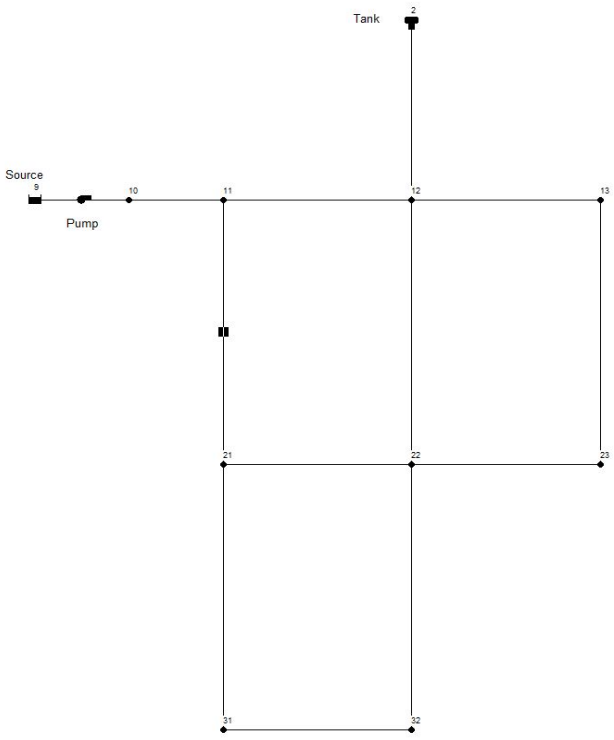
$$D(k) = MNF(k) - min\{MNF(i,w)\}, \;\; where \;\; i \in \{k-1-w, ..., k-1\} \tag{2.8}$$

Assume T is a percentage threshold. An anomaly flag is raised when Equation 2.9 is satisfied.

$$D(k) > T \times min\{MNF(i,w)\} \tag{2.9}$$



(a) Hanoi WDS



(b) Net1 WDS

Figure 2.8: WDS in LeakDB

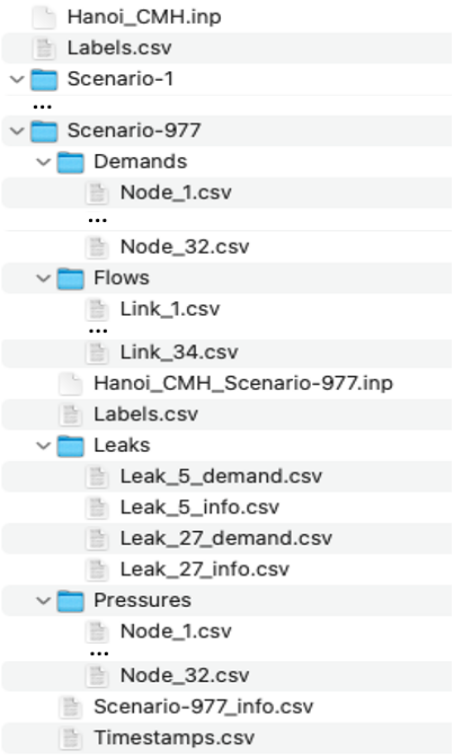


Figure 2.9: Overview of the files in LeakDB using Hanoi WDS as example.

Assume the true positive is TP; false positive is FP; true negative is TN; false negative is FN. The MNF detector is evaluated by the three metrics in Equation 2.10, Equation 2.11 and Equation 2.12. As this research does not work on the early warning, the early detection in [86] is not discussed.

$$R_{TP} = TP/(TP + FN) \tag{2.10}$$

$$R_{TN} = TN/(FP + TN) \tag{2.11}$$

$$F_{tp,fp} = 2TP/(2TP + FP + FN) \tag{2.12}$$

Three threshod T = $\{10\%, 20\%, 30\%\}$ and w = 10 days are tested. The results are presented in Table 2.9.

Method	Advantage	Disadvantage
Statistical-Based Method	If the data distribution or regression model is correctly identified, the method applied and analytical data's flow can be interpreted with proven statistical theories. It can be used as unsupervised learning.	It requires the data is in a specific distribution or regression model. It requires large data to tune the model's hyperparameters.
Nearest Neighbor-Based	It is easy to integrate with distance models. It does not require the data in any form of distribution model.	The detection accuracy highly depends on the quality of both training and test data sets. It is sensitive to selecting the methods for distance measurement, while finding the correct distance measurement method is not easy. Most methods require high memory and computation power in large data sets computation.
Clustering-Based Method	It can be used as incremental models. It can be fast in testing results as clusters are identified. It can be used as unsupervised learning.	The detection accuracy highly depends on if the correct clustering method is selected. The detection accuracy highly depends on the quality of the data.
Classification-Based	It is helpful to classify the sample as normal or abnormal behaviors. It is efficient to deliver the result as the model is pre-trained.	It requires proper label data sets for classes that are hard to acquire.
Deep Learning-Based	It allows automatic feature extraction. It is easy to uncover the relations among unstructured data. It does not require well-labeled data. It is efficient to deliver the result as the model is pre-trained.	It requires large data sets to train the model. It has many parameters to be tuned in order to solve overfitting. It does not have standard architecture. It creates black boxes in how a neural network arrives at a particular solution.

Table 2.2: Summary of time series anomaly detection algorithm

Corpus	WDS	Contextual Outliers	Collective Outliers	Anomalies Label	Multivariate
BATDAL [83]	Yes	Yes	Yes	Yes	Yes
Numenta Anomaly Benchmark [2]	No	Yes	No	Yes	Yes
Air Quality [21]	No	Yes	Yes	Yes	Yes
EEG Database [10]	No	Yes	Yes	Yes	Yes
LeakDB [86]	Yes	Yes	Yes	Yes	Yes

Table 2.3: Comparison of the public datasets

Dataset Label	Time Period	Data Resolution	Contains Attacks
1	1 Year	Hourly	No
2	6 Months	Hourly	7 Attacks
3	3 Months	Hourly	7 Attacks

Table 2.4: Overview of the BATADAL datasets

	Dataset 1 and Dataset 2			Dataset 2			Dataset 3		
Model	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision
AE	/	/	/	0.704	0.608	0.835	0.715	0.602	0.881
XGBoost	0.706±0.039	0.555±0.046	0.976±0.028	/	/	/	0.149	0.081	0.943
LightGBM	0.707±0.057	0.565±0.070	0.951±0.019	/	/	/	0.097	0.052	0.840

Table 2.5: Performance of AE, XGBoost, and LightGBM on the BATADAL datasets [82]

	1D CNN	AE	VAE	PCA Reconstruction	AE Frequency
Attack	7 (1 fp)	7	7 (3 fp)	7	7
F1	0.833	0.919	0.783	0.875	0.937

Table 2.6: Performance of neural network on the BATADAL datasets [48]

Network	Number of Scenarios	Time Period of Each Scenario	Data Resolution	Anomaly Label
Hanoi	1000	1 Year	30 Minutes	Yes
Net1	1000	1 Year	30 Minutes	Yes

Table 2.7: Overview of Hanoi and Net1 WDS in LeakDB

Network	Type of Sensors	Number of Sensors
Hanoi	Node Pressure	32
Hanoi	Node Demand	32
Hanoi	Link Flow	34
Net1	Node Pressure	11
Net1	Node Demand	11
Net1	Link Flow	13

Table 2.8: Type of sensors in Hanoi and Net1 WDS

Detector	Score		
	R_{TP}	R_{TN}	$F_{tp,fp}$
MNF 10%	68.98	66.96	50.01
MNF 20%	39.90	99.58	56.49
MNF 30%	32.87	99.91	49.36

Table 2.9: Performance of MNF detectors on the LeakBD dataset of Hanoi WDS [86].

Chapter 3

Anomaly Detection Using Ensemble Method

This chapter presents an ensemble anomaly detection method. The state-of-the-art AE-based anomaly detection algorithm, referred to in Chapter 2.3.2, is a semi-automated method. Specifically, the user requires domain knowledge (for example, hydraulic modeling of WDN) and needs to manually analyze the AE reconstruction error to get good detection and localization results. To achieve data-driven detection and localization with minimum intervention from the user, we propose a matrix profile and AE-based ensemble method. The matrix profile method was initially proposed for the UTS, and we adopt this method for the MTS. AE is then introduced to remove false-positive detections.

3.1 Matrix Profile: Preliminaries

The matrix profile has two major components: distance profile and profile index. The distance profile is a vector storing the minimum z-normalized Euclidean distance. The z-normalized Euclidean distance $d_{x,y}$ of two time series subsequences $T_{x,w}$ and $T_{y,w}$ can be calculated as followed:

$$d_{x,y} = \sqrt{2w(1 - \frac{Q_{x,y} - w\mu_x\mu_y}{w\sigma_x\sigma_y})} \tag{3.1}$$

Where w is the length of subsequence, μ_x is the mean of $T_{x,w}$, σ_x is the standard deviation of $T_{x,w}$, μ_y is the mean of $T_{y,w}$, σ_y is the standard deviation of $T_{y,w}$, and $Q_{x,y}$ is the dot product of $T_{x,w}$ and $T_{y,w}$ [96]. The profile index contains the index of its first nearest neighbor, which is the location of its most similar subsequence. The calculation of matrix profile for time series is illustrated in Figure 3.1, and the steps involved are presented below.

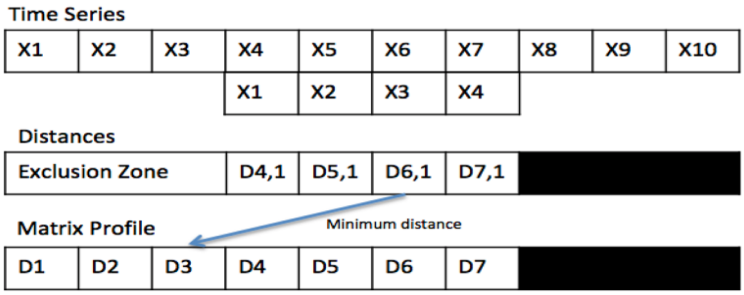


Figure 3.1: Calculation of matrix profile using $T_{1,4}$ as example.

1. Given a time series T with length N, $T_{x,w}$ is a subsequence of length w starting from position x in T.

2. Use sliding window to compare the $T_{x,w}$ with the subsequence of T having the same window size.

3. Compute and collect the distances in each comparison.

4. Set exclusion zone to avoid distance from the self-match.

5. Record the minimum distance and the first nearest-neighbor index.

6. Repeat above steps for (N-w+1) times.

The exclusion zone prevents trivial matches as the small distance contributed by $T_{x,w}$ itself is not informative. The exclusion zone needs to be expanded if the nearby subsequence is likely to be highly similar to $T_{x,w}$.
Motifs and discords discovery are two primary applications for utilizing matrix profile. Motifs are subsequences, which appear multiple times within the time series with a similar shape. Based on some distance measures, motif candidates are selected as subsequences with low distances to each other and ranked. Ranking criteria is either the number of occurrences or the minimum distance between pairs of motif instances. The motifs are called top-frequent motifs in the first case and range motifs in the second one. In all cases, trivial matches are not taken into consideration. The fastest known exact algorithm for computing time series motifs is the MK algorithm [84] [61]. A time-series discord is the subsequence that has the maximum distance to its nearest neighbor. It is usually the sequences with the most unusual behavior within the time series. Discord discovery is then utilized for anomaly detection.
The motifs and discords are illustrated in Figure 3.2(a) and Figure 3.2(b) with the help of the BATADAL dataset. The motifs are obtained from the matrix profile by looking at the local minimums, while the discords are collected from the matrix profile by looking at the

local maximums. The only hyperparameter defined in motifs and discords discovery is the window length w for the sliding window.

3.2 Autoencoder: Preliminaries

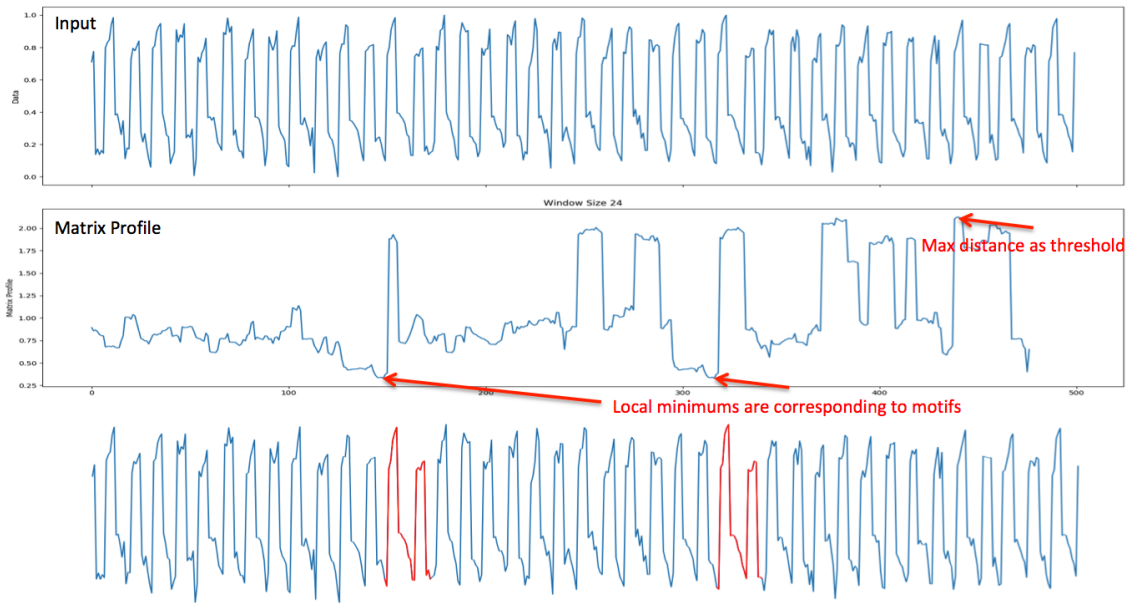
AE is a neural network that aims to transform inputs into outputs with the least possible distortion [9]. AE consists of four main parts:

- encoder for reducing dimensions and compressing the input,
- bottleneck containing the lowest possible dimension of the input,
- decoder for reconstructing the data,
- reconstruction loss representing the decoder's performance.

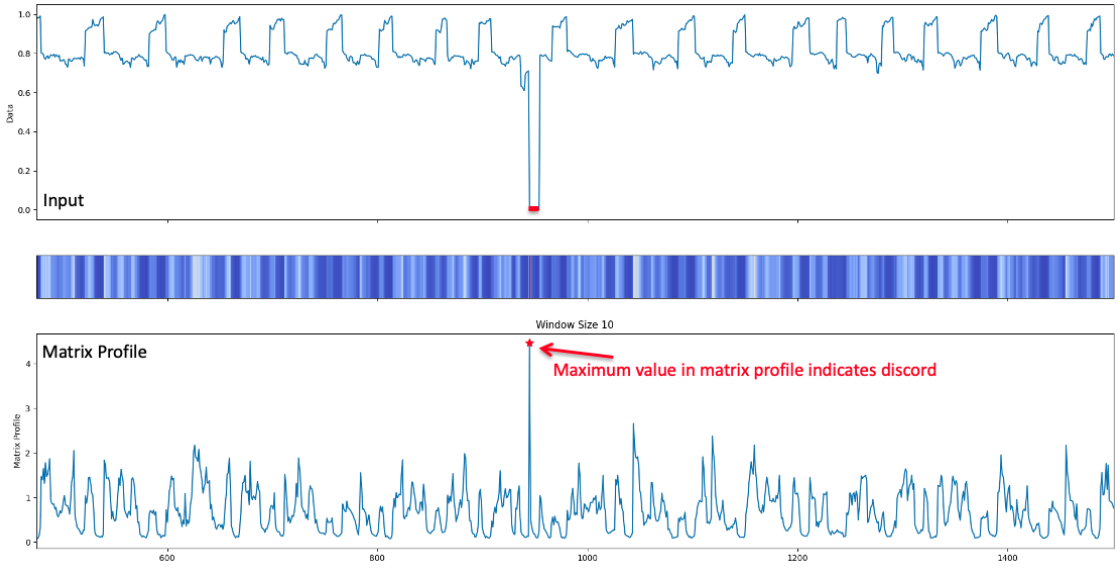
The AE defines the hidden layers. Then the encoder takes the inputs and converts them to codes, while the decoder converts the codes and reconstructs the inputs. The ratio of input size to bottleneck size is the compression factor. Higher values for the compression factor lead to coarser reconstructions. The general AE architecture is presented in Figure 3.3. To minimize the average reconstruction error, the optimization process is carried out with the Adam algorithm, while early stopping is added to prevent the models from overfitting [82]. Assume input is vector i. i is mapped to hidden representation h. $h = m(i)$ where the encoder network parametrizes the encoder function m by at least one layer of non-linearity. The hidden representation h is finally mapped to output vector o. $o = n(h)$ where function n is parametrized by the decoder network with at least one layer of non-linearity. The cost function of Equation 3.2 is introduced to optimize the parameters.

$$\Gamma(i, g(f(i))) = \parallel o - i \parallel^2 \tag{3.2}$$

A constraint limiting the autoencoder's representational capacity is generally added to optimize the cost function. It reduces the chance of learning invalid identity functions. The constraint can be implemented via approaches like limiting the latent code dimensionality in architecture or using sparsity constraint.



(a) Motif discovery



(b) Discord discovery

Figure 3.2: Motif and discord discovery by matrix profile. The x-axis is the time, and the y-axis is the distances calculated by matrix profile.

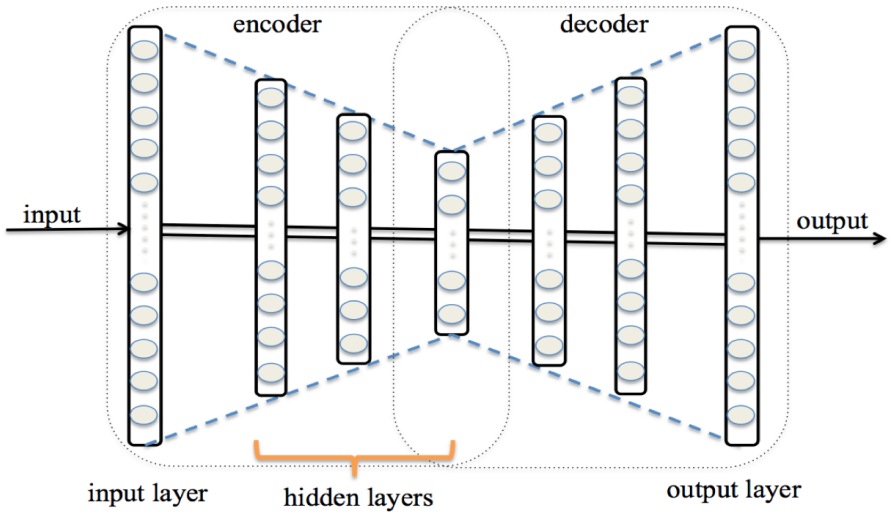


Figure 3.3: Time series anomaly detection using autoencoder method.

3.3 Workflow

In this section, we describe the proposed workflow for anomaly detection. The workflow for the detection model is presented in Figure 3.4. The model has two functions: detection and localization. Detection includes two phases: the training phase and the testing phase. Dataset 1 is used for training, while dataset 2 and dataset 3 are used for testing. The description of each dataset can be found in Chapter 2.3.2. The training phase focuses on feature extraction, and the testing phase utilizes the features to identify anomalies. After detection is conducted, the model identifies the sensors potentially affected by the anomalies, which are the simulated cyber-attacks in the experiment. Next, a detailed description of various blocks in Figure 3.4 is presented.

3.3.1 Signal Preprocessing

The steps involved in signal preprocessing are presented in Figure 3.5(a). Sensor signals are categorized into three types.

- The first type are linear signals. They are data from sensors indicating the status of the pump and the flow of the pump. The specific sensor list is *S_PU1*, *S_PU3*, *S_PU5*, *S_PU6*, *S_PU9*, *S_PU11*, *F_PU3*, *F_PU5*, *F_PU6*, *F_PU9*, and *F_PU11*. For these sensors, the spikes are potentially caused by abnormal behavior. The first type of signal is named signal L in the report. The plot of partial data collected from *S_PU1* is presented in Figure 3.6(a).

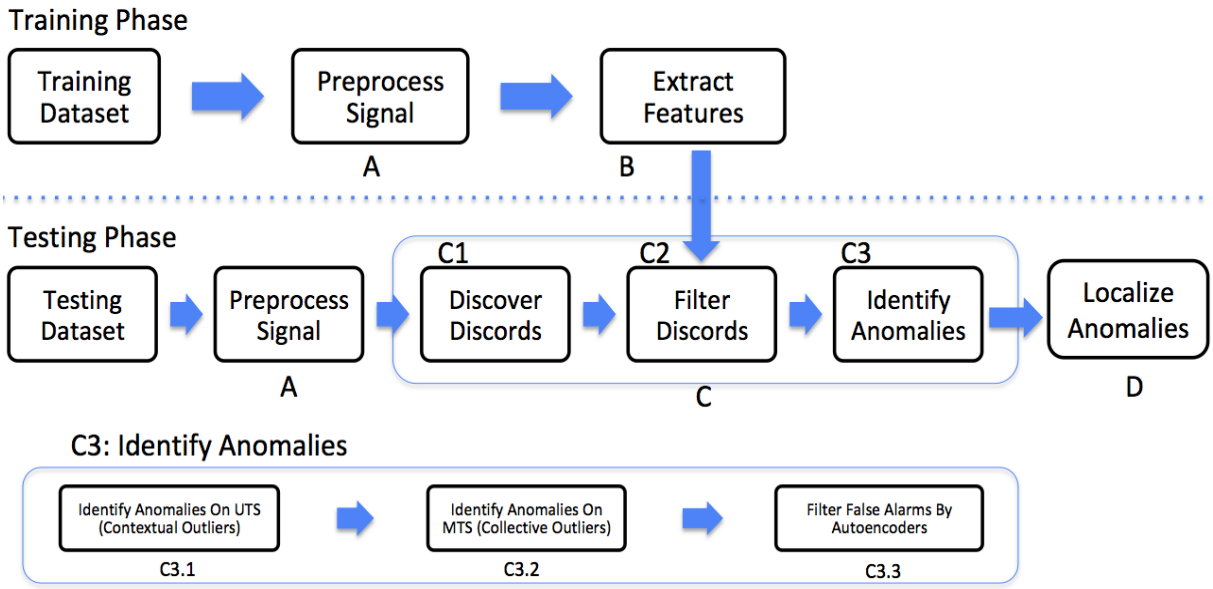
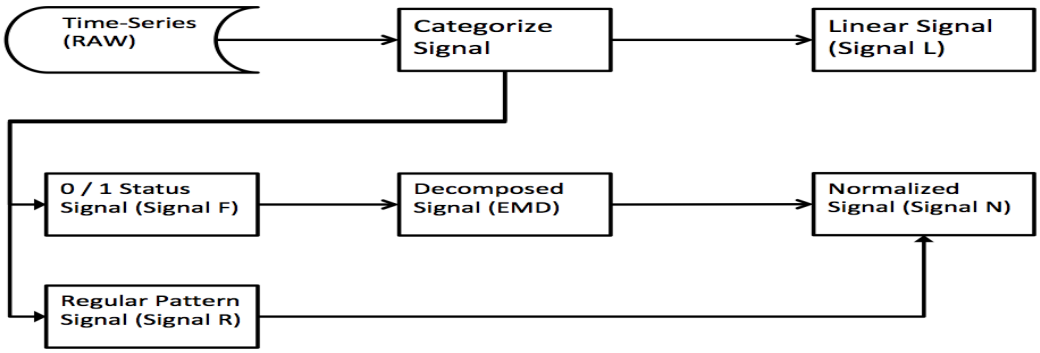


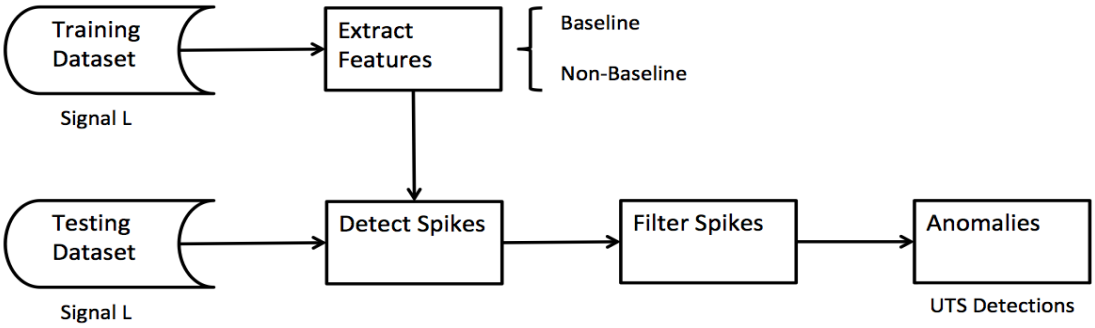
Figure 3.4: Workflow of Anomaly detection using the ensemble method

- The second type of signal is from sensors recording pump and valve status. They are *S_PU2*, *S_PU4*, *S_PU7*, *S_PU8*, *S_PU10*, and *S_V2*. As the sensor's data value is either 0 or 1, the frequency of the status change can reflect normal operations from another perspective. The EMD method described in Chapter 2.1.5 decomposes the signal and captures the change in status. As IMF_2 already has a certain periodicity and relatively regular pattern, it is selected to be the input signal for the next steps. The order of the computational complexity of the EMD is equivalent to FFT [87]. Therefore, the time complexity for extracting all IMFs is given as $O(nlogn)$. The second type of signal is named signal F in the report. The plot of partial data collected from S_PU2 is presented in Figure 3.6(c), and the plot of its decomposed data is presented in Figure 3.6(d).

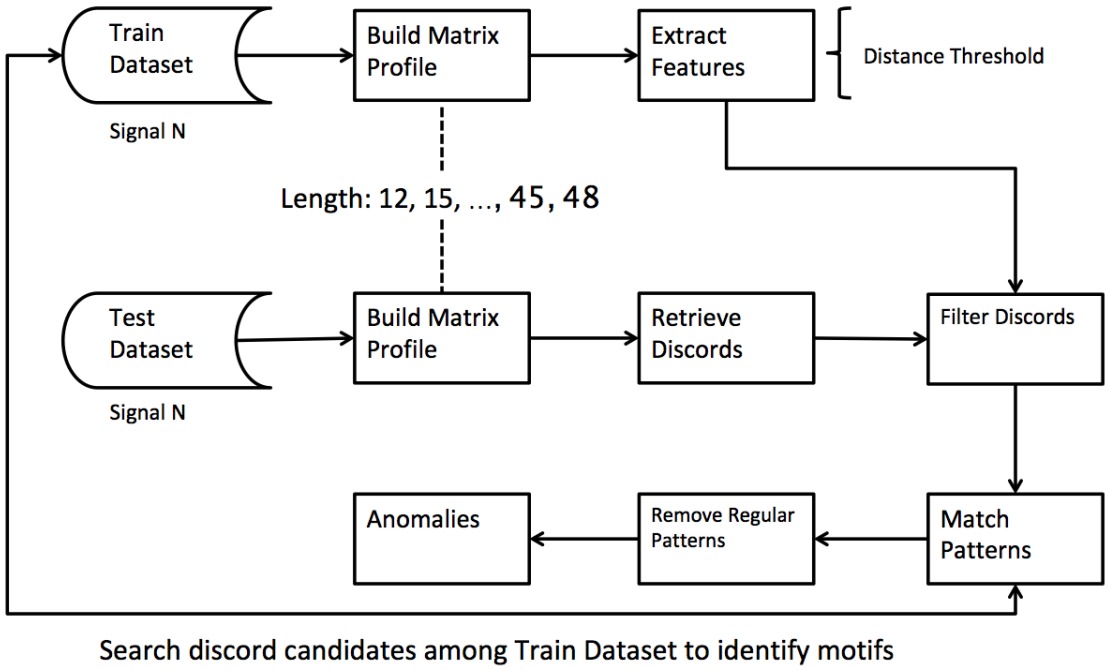
- The third type of signal is from the rest of the sensors. They have regular patterns to reflect normal operations in the network. The third type of signal is named signal R in the report. The plot of partial data collected from F_PU1 is presented in Figure 3.6(b). Both decomposed signal L and signal R are normalized before going to the next stage. The normalization scales their values to the given range between zero and one.



(a) Preprocessing of the sensor signal

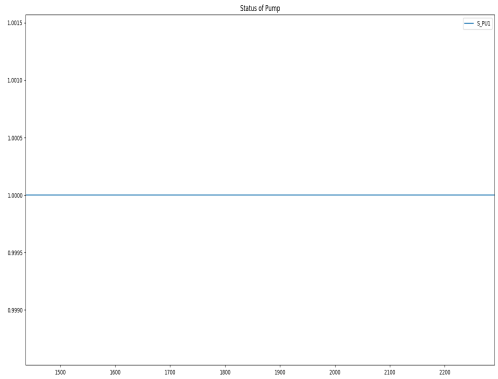


(b) Workflow of anomaly detection for the linear signal

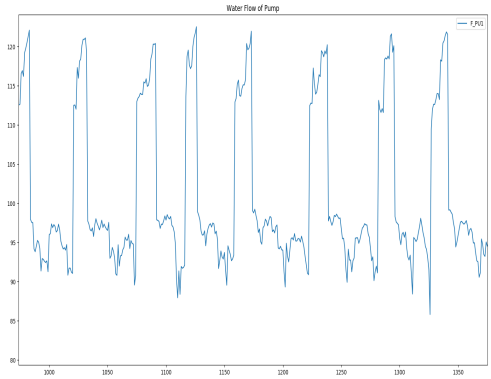


(c) Workflow of anomaly detection for the regular pattern signal

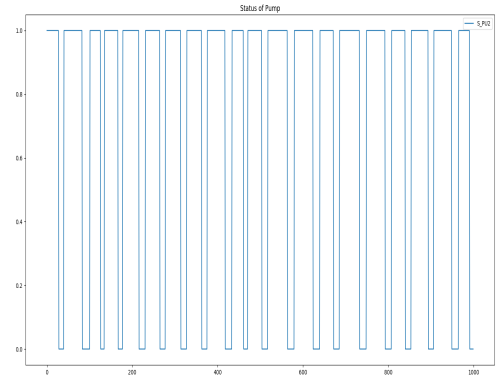
Figure 3.5: Workflow of anomaly detection using matrix profile



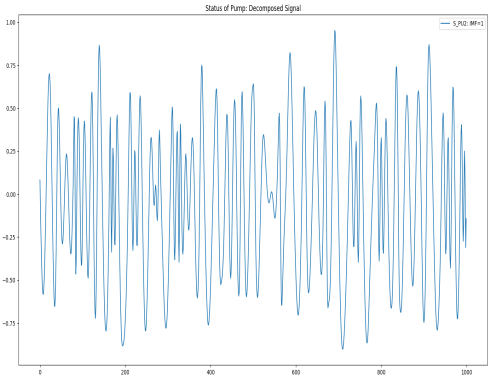
(a) Signal L



(b) Signal R



(c) Signal F



(d) Signal F (Decomposed)

Figure 3.6: Three types of sensor signals: Signal L, Signal R and Signal F. Signal F is decomposed by EMD.

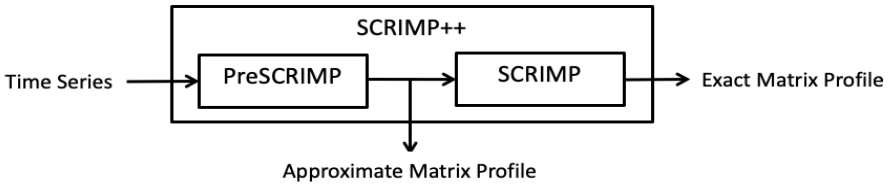


Figure 3.7: SCRIMP++

3.3.2 Feature Extraction

To detect anomalies in signal L, the model finds the baseline of the signal and defines the values not on the baseline as spikes, which are potential anomalies for sensors. The signal's baseline and the max continuous hours of the spikes of each sensor in the training dataset are then recorded.

To detect anomalies in signal F and signal R, the model implements the matrix profile to retrieve the distance thresholds used in discord identification in the testing stage. In this thesis, among the frequently used matrix profile algorithms, SCRIMP++ is selected as the solution since it combines the best features from STOMP and STAMP. The SCRIMP++ Algorithm can be divided into two parts as shown in Figure 3.7. The first part is an ultra-fast algorithm, called PreSCRIMP, for preprocessing signals. It gives a precise approximation of the matrix profile. The second part is an anytime algorithm, called SCRIMP, for refining the approximated matrix profile to get the final result. Both PreSCRIMP and SCRIMP can be interrupted at any time. Therefore, SCRIMP++ is an anytime algorithm and has the complexity of $O(n^2)$ [95]. The algorithms of SCRIMP and PreSCRIMP are presented in Algorithm 1 and Algorithm 2.

The signal F and signal R in the training dataset are built using the matrix profile. As the anomalies can last for different lengths of time, we define an array to scan over the sequences. The array is a collection of window lengths of $\{12, 15, \ldots, 45, 48\}$. Assume DT is the discord threshold, T is the input time series of the training dataset, w is the subsequence length. The discord distance threshold can be generated by Equation 3.3 and Equation 3.4.

$$MP(P(w), X(w)) = SCRIMP++(T, w) \tag{3.3}$$

$$DT(w) = max\{d \in P(w)\} \tag{3.4}$$

where w is the window length in matrix profile calculation and w has the values of 12, 15, ..., 45, 48.

Algorithm 1: The SCRIMP Algorithm

Input : A time series T and a subsequence length w
Output: Matrix profile P and matrix profile index X of T

$n \leftarrow Length(T)$;
$\mu, \sigma \leftarrow ComputeMeanStd(T, w)$;
$P \leftarrow infs, X \leftarrow ones$;
$Orders \leftarrow RandPerm(w/4 + 1 : n - w + 1)$;
for $y \in Orders$ **do**
    for $x \leftarrow 1$**to** *n-w+2-y* **do**
        if $x = 1$ **then**
            $q \leftarrow DotProduct(T_{1,w}, T_{y,w})$;
        else
            $q \leftarrow q - t_{x-1}t_{x+y-2} + t_{x+w-1}t_{x+y+w-2}$;
        end if
        // Line 7 -11 calculating dot product [96].
        $d \leftarrow CalculateDistance(q, \mu_x, \sigma_x, \mu_{x+y-1}, \sigma_{x+y-1})$;
        if $d < P_x$ **then**
            $P_x \leftarrow d, X_x \leftarrow x + y - 1$;
        end if
        if $d < P_{x+y-1}$ **then**
            $P_{x+y-1} \leftarrow d, X_{x+y-1} \leftarrow x$;
        end if
    end for
end for

Algorithm 2: The PreSCRIMP Algorithm

Input : A time series T, a subsequence length w and a sampling interval s
Output: The running matrix profile P and matrix profile index X of T

1 $n \leftarrow Length(T), P \leftarrow infs, X \leftarrow ones$;
2 $\mu, \sigma \leftarrow ComputeMeanStd(T, w)$;
3 **for** $x \leftarrow RandPerm(1 : s : (n - w + 1))$ **do**
4 $seq \leftarrow T_{x,w}$;
5 $D \leftarrow MASS(T, seq)$;
6 $P, X \leftarrow ElementWiseMin(D, P, x)$;
7 $P_x, X_x \leftarrow min(D)$;
8 // Line 4 - 7 evaluate the distance profile using MASS algorithm [88], an ultra-fast way for similarity search.
9 $y \leftarrow X_x$;
10 $q \leftarrow CalculateDotProduct(P_x, \mu_x, \sigma_x, \mu_y, \sigma_y), q' \leftarrow q$;
11 **for** $k \leftarrow$ 1**to** *min(s-1, n-w+1- max(x,y))* **do**
12 $q \leftarrow q - t_{x+k-1}t_{y+k-1} + t_{x+k+w-1}t_{y+k+w-1}$;
13 $d \leftarrow CalculateDistance(q, \mu_{x+k}, \sigma_{x+k}, \mu_{y+k}, \sigma_{y+k})$;
14 **if** $d < P_{x+k}$ **then**
15 $P_{x+k} \leftarrow d, X_{x+k} \leftarrow y + k$;
16 **end if**
17 **if** $d < P_{y+k}$ **then**
18 $P_{y+k} \leftarrow d, X_{y+k} \leftarrow x + k$;
19 **end if**
20 **end for**
21 $q \leftarrow q'$;
22 **for** $k \leftarrow$ 1**to** *min(s-1, x-1, y-1)* **do**
23 $q \leftarrow q - t_{x-k+w}t_{y-k+w} + t_{x-k}t_{y-k}$;
24 $d \leftarrow CalculateDistance(q, \mu_{x-k}, \sigma_{x-k}, \mu_{y-k}, \sigma_{y-k})$;
25 **if** $d < P_{x-k}$ **then**
26 $P_{x-k} \leftarrow d, X_{x-k} \leftarrow y - k$;
27 **end if**
28 **if** $d < P_{y-k}$ **then**
29 $P_{y-k} \leftarrow d, X_{y-k} \leftarrow x - k$;
30 **end if**
31 **end for**
32 **end for**
33 **return** *P, X*;

3.3.3 Anomalies Detection

To detect anomalies from signal L, the model picks up values that deviate from the baseline and counts the time duration of non-baseline values. If the consecutive hours are more than the max non-baseline consecutive hours from the training dataset of the same sensor, they are tagged as anomaly candidates.

To detect anomalies from signal F and signal R, the model builds the matrix profile using Equation 3.5.

$$MP(P_{x,w}(w), X_{x,w}(w)) = SCRIMP++(T_{x,x+s}, w) \tag{3.5}$$

Where T is the time series of testing datasets; x is the start index of testing subsequence; s is the size of testing subsequence; and w has the values 12, 15, ..., 45, 48. Then it finds the top K discords for each profile, where K has the value of 29. A large K is used to identify all potential discords. For this case, when K has the value of 29, it assumes that the utility receives 29 attacks in three months, a number of attacks which has a very low probability of occurrence. Then in the next step, the model filters the discords by removing the discord with a distance of less than $DT(w)$, which is given in Equation 3.4 and obtained from the training dataset. The remaining discords are searched along with the training dataset via Equation 3.6. The distance of the most matched subsequence between the discord and the training dataset is calculated via Equation 3.7.

$$MP(P_{y,w}, X_{y,w}) = SCRIMP++(T_{x,w}, w, T_y) \tag{3.6}$$

$$DM(w) = min\{d \in P_{y,w}\} \tag{3.7}$$

Where x is the start index of testing subsequence, w is the window length of the pattern, T_y is the training dataset. If $DM(w)$ is smaller than the $DT(w)$, the discord is classified as the regular pattern and removed. Discords that are not filtered out by this regular pattern filter are potential anomalies.

Since the physical processes are interdependent within the WDS, an anomaly, like a cyber-attack, affects multiple sensors and is reflected on the data collected from the sensors. Thus, the anomalies picked up are collective outliers. After detecting potential anomalies from each sensor, the anomalies are integrated, and only anomalies detected by at least two sensors are considered as anomalies.

In order to find the discords and match them with the motifs from the training dataset, the length of time defined for the calculation matrix profile is up to 48 hours. As a result, anomaly detection using matrix profile does not identify the attack start time and end time accurately. To solve this problem, we integrate the matrix profile with autoencoders. We use

autoencoders to detect the start and end times of the anomaly. Specifically, we define that the attacks only happen in the window, W, where the matrix profile model has consecutive hours of positive detection; inside W, the attack starts from the first autoencoder positive detection and ends within the last autoencoder positive detection.

3.3.4 Localization

As the model analyzes univariate time-series in the first detection stage, localization is automated, and the result is derived from the UTS results from each sensor.

3.4 Results

The model is built based on the workflow described in Section 3.1. We evaluate the results by the following performance metrics.

- Recall: measures the ability to correctly classify positive observations.
- Precision: measures the ability to prevent false alarms.
- F1: combines recall and precision to give the same importance to both metrics.
- Accuracy: measures the ability to correctly predict the observations over the total observations

BATADAL has three datasets: Dataset 1, Dataset 2, and Dataset 3. We trained the model on Dataset 1 and used the other two datasets for testing. To simulate the data coming live, subsequences of size 2089 are selected each time to assess the model's performance. Then dataset 2 is tested with subsequences starting from index n = 0, 1, 2088, and dataset 3 is tested with subsequences starting from index n = 0. By taking the average of the metrics from each test result, Table 3.1 is constructed and presents the performance of the model.

	Accuracy	F1	Precision	Recall
Dataset 2	0.854	0.580	0.414	0.975
Dataset 3	0.781	0.632	0.470	0.966

Table 3.1: Results of anomaly detection using matrix profile based on accuracy, F1, precision and recall

Taking the starting index of 0 and 2088 for dataset 2, and starting index of 0 for dataset 3, the results are drawn in Figure 3.8 to view if the model detects the cyber-attacks. The

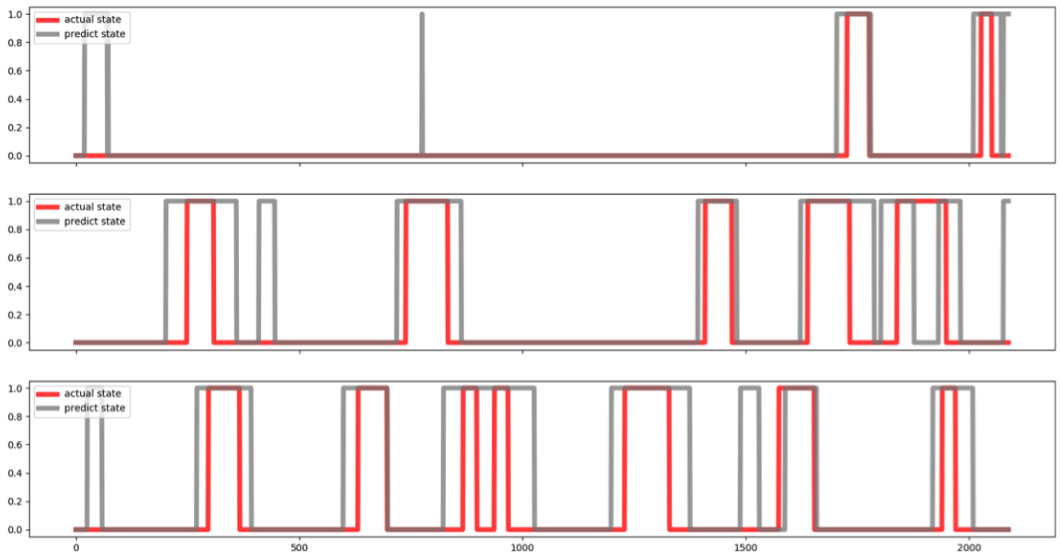


Figure 3.8: Result for anomaly detection using matrix profile. Red color indicates the ground truth, i,.e, presence of anomaly and gray color indicates predicted anomalous events.

ground truth and the detected anomalies are highlighted with red and gray colors, respectively.

The proposed matrix profile method is a semi-supervised anomaly detection algorithm. Therefore, we select the AE-based semi-supervised anomaly detection algorithm proposed in [82] as the benchmark. Both models are trained with Dataset 1, which does not contain the attack, and then tested on Dataset 2 and Dataset 3. In contrast, the autoencoder is capable of multivariate anomaly detection, while the matrix profile model analyses the time series dataset from each sensor and integrates the results by following the concept of collective outliers in the network. The matrix profile model achieved better recall statistics. It detected 13 attacks thoroughly and partially detects one attack. However, to find the discords and match them with the motifs from the training dataset, the window length w defined for the calculation matrix profile was up to 48 hours. As a result, the detected start time and end time of the events were not accurate. False alarms were raised, especially before the event start and after the event end. Consequently, this lead to lower precision compared to the AE-based approach.

We observed the advantages of autoencoder when we conducted a literature survey and review on benchmark algorithms. To improve precision, we propose an ensemble method to integrate the matrix profile with autoencoder. The benchmark uses the autoencoder to detect anomalies, while autoencoder is used for filtering false alarms in the ensemble method. The autoencoder corrects the event's starting time and ending time. Specifically, we define

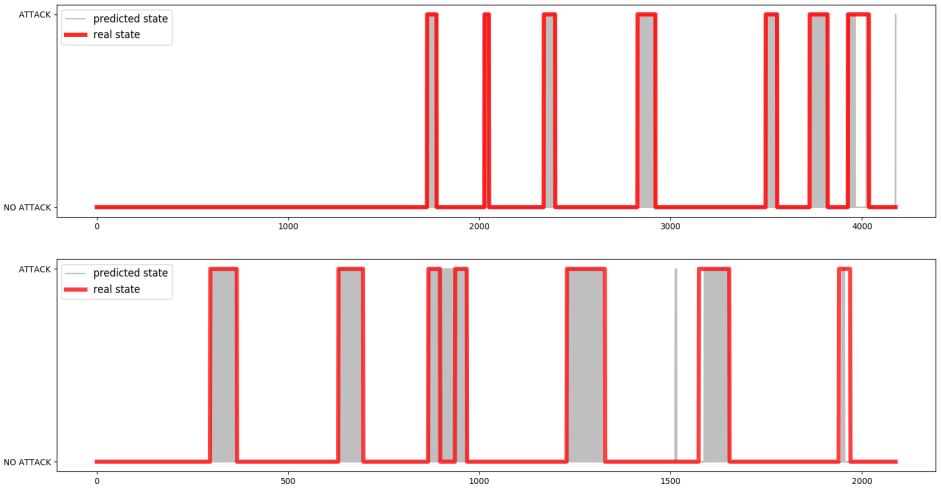


Figure 3.9: Result of anomaly detection using ensemble method

that the attacks only happen in the window, W, where the matrix profile model has consecutive hours of positive detection; inside W, the attack starts from the first positive detection by the autoencoder and ends with the last positive detection by the autoencoder.
Taking 43 as inputs, 3 as hidden layers in the encoder and decoder, and 2.5 as compression factor to set up the autoencoder, dataset 2 and dataset 3 are tested. The ensemble model gets better performance metrics comparing with benchmarking algorithms using AE directly, as shown in Table 3.2. Figure 3.9 illustrates the accuracy of the proposed ensemble approach.

	Dataset 2				Dataset 3			
Model	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall
Ensemble Model	0.974	0.878	0.987	0.790	0.955	0.882	0.890	0.875
AE [82]	/	0.704	0.835	0.608	/	0.715	0.881	0.602

Table 3.2: Result of anomaly detection using ensemble method based on accuracy, F1, precision and recall

As the detection model starts with the analysis of univariate signals, to localize the attack, the anomalies observed by each sensor can be retrieved to provide information on the potential location of security concerns. Taking the starting index of 0 and 2088 for dataset 2 and starting index of 0 for dataset 3, the localization results for each attack are presented in Table 3.3. Figure 3.10 gives an example of sensor F_V2.

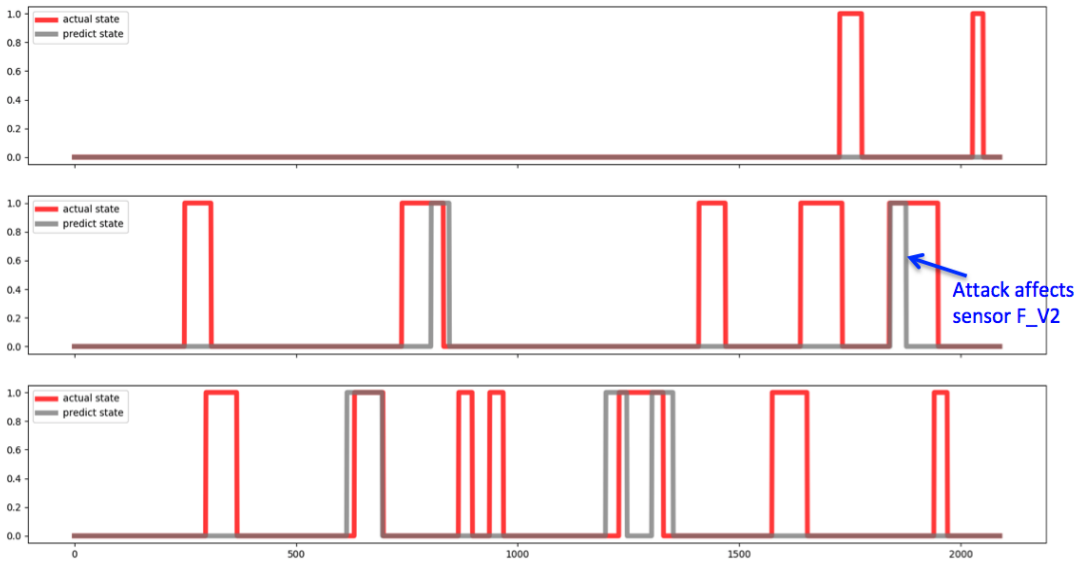


Figure 3.10: Localization of the anomaly detection based on sensor F_V2

3.5 Conclusion

The proposed ensemble model uses matrix profile and autoencoders to detect cyber-attacks in the BATADAL dataset. The model reveals contextual anomalies. Recalling the interdependence of the physical process in WDS, the model assumes that anomalies affect multiple sensors in the network and detect cyber-attacks as collective anomalies. It finally gets the average performance metrics: accuracy of 0.9645, precision of 0.9385, recall of 0.8325, and F1 score of 0.8800. Additionally, localization results are auto-generated as part of anomaly detection in univariate analysis.

Label	Start Time (dd/mm/yyyy hh)	End Time (dd/mm/yyyy hh)	Sensors Detected Anomalies
1	13/09/2016 23	16/09/2016 00	F_PU10, P_J280, P_J302, P_J307, S_PU11, F_PU11, L_T6
2	26/09/2016 11	27/09/2016 10	P_J280, P_J317
3	09/10/2016 09	11/10/2016 20	L_T1, F_PU1, F_PU2, P_J280, P_J269, S_PU10, P_J14
4	29/10/2016 19	02/11/2016 16	L_T1, P_J14, P_J269, F_PU1, F_V2, P_J280, P_J300, P_J289, P_J302, P_J307, P_J422
5	26/11/2016 17	29/11/2016 04	S_PU6, S_PU7, P_J415, P_J280, F_PU6
6	06/12/2016 07	10/12/2016 04	F_PU7, P_J280, F_PU6, S_PU6
7	14/12/2016 15	19/12/2016 04	F_V2, P_J280, P_J317, S_PU11, P_J280
8	16/01/2017 09	19/01/2017 06	P_J269, P_J256, F_PU1, P_J280, L_T1
9	30/01/2017 08	02/02/2017 00	L_T1, L_T2, F_PU1, F_PU2, F_V2, P_J280, P_J269, P_J300, P_J289, P_J14, P_J422
10	09/02/2017 03	10/02/2017 09	F_PU2, P_J280, F_PU3, S_PU3, L_T2
11	12/02/2017 01	13/02/2017 07	F_PU1, P_J280, P_J269, F_PU3, S_PU3, F_PU2, S_PU1, L_T1
12	24/02/2017 05	28/02/2017 08	F_V2, P_J14, P_J422, P_J280, P_J300, P_J289, P_J256
13	10/03/2017 14	13/03/2017 21	L_T6, P_J302, P_J280
14	25/03/2017 20	27/03/2017 01	P_J280, P_J415

Table 3.3: Localization of the anomaly detection using matrix profile

Chapter 4

Anomaly Detection Using Boosting Method

This chapter presents three boosting-based anomaly detection methods. Chapter 3 focuses on the semi-supervised learning method to detect anomalies because anomalies are rare, and labeling anomalies is quite challenging. However, supervised learning allows a more accurate decision boundary to be set in differentiating classes [30]. We can utilize supervised detection when the anomaly labels are well collected and maintained. In the literature survey, we find that using boosting algorithms can achieve relatively higher accuracy in anomaly detection on MTS [22] [59] [29]. Taormina and Galelli (2018) in [82] also select XGBoost and LightGBM as benchmark algorithms, while steps of applying both methods are not described in the paper. Therefore, this research decides to have a comparative study on boosting methods and utilize boosting algorithms to detect anomalies. Comparing with XGBoost [16] and LightGBM [42], this research finds CATBoost [26] has better performance among the boosting algorithm on BATADAL datasets. In the next sections, we present anomaly detection using the three boosting methods.

4.1 Boosting: Preliminaries

Boosting is a machine learning algorithm that can be used to reduce bias in supervised learning [76]. It trains multiple weak classifiers and finally combines them into a strong classifier with weights. A weak classifier generally refers to a classifier whose classification results are only slightly better than random classification. [78].

Gradient Boosting Decision Tree (GBDT) is an essential and popular boosting algorithm. GBDT is an ensemble model and takes the classification and regression tree (CART) as the base classifier. CART uses the least square error to select the optimal segmentation

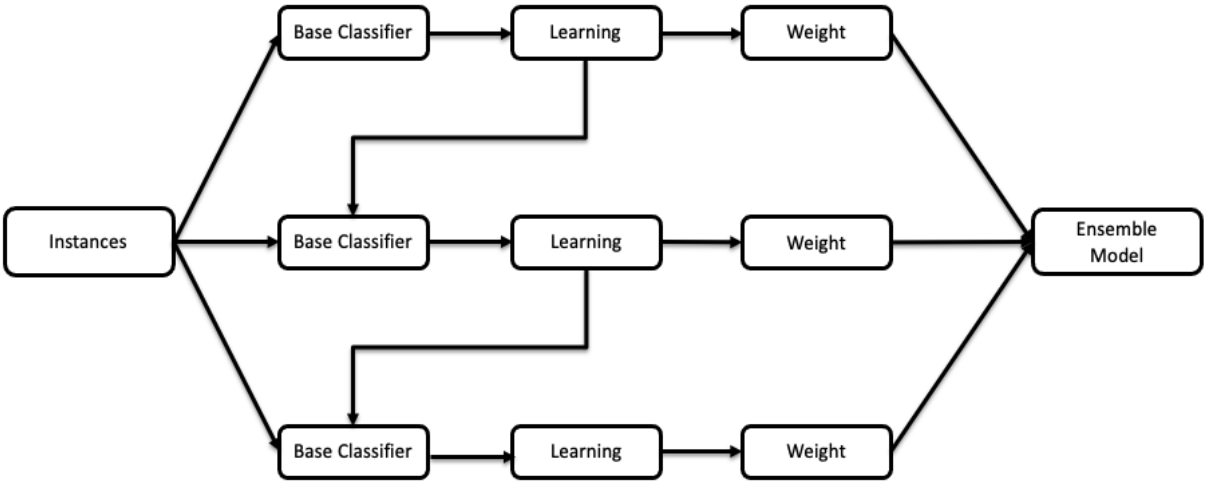


Figure 4.1: Workflow of the Gradient Boosting Decision Tree

feature and segmentation point during classification. Figure 4.1 presents the workflow of the GBDT algorithm. GBDT goes through multiple rounds of iteration, each iteration produces a weak classifier, and each classifier is trained based on the residual of the previous round of classifiers. Because the training process is to continuously improve the accuracy of the final classifier by reducing bias, the requirements for weak classifiers are low variance and high bias. The final classifier is the weighted summation of the weak classifiers obtained in each round of training.

4.1.1 XGBoost

XGBoost is an effective implementations of GBDT, which was proposed in [16] in 2016. The big difference between XGBoost and traditional gradient tree boosting algorithms is the definition of the objective function. Define $T = \{x_i, y_i\}$ $(|T| = n, x_i \in R^m, y_i \in R)$ for a given data set with n examples and m features. Assume K additive functions are used to predict the output.

$$\hat{y}_i = \theta(x) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \Phi \tag{4.1}$$

where Φ is the space of CART. Assume N is the leaves' number, f_k is an independent tree structure with leaf weights w. We use the decision rules in the trees to classify input data into the leaves and calculate the final prediction by summing up the score, w, in the corresponding leaves. To learn the set of functions used in the model, we minimize the following regularized

objective.

$$\Gamma(\theta) = \sum_i z(\hat{y}_i, y_i) + \sum_k \Upsilon(f_k), \;\; where \;\; \Upsilon(f) = \varphi N + 1/2\lambda \| \; w \; \|^2 \tag{4.2}$$

Here z is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i. The second term Υ penalizes the complexity of the mode. The additional regularization term helps to penalize large weights to avoid over-fitting. Intuitively, the regularized objective tends to select a model employing predictive and straightforward functions. When the regularization parameter is zero, the objective falls back to traditional gradient tree boosting.

4.1.2 LightGBM

GBDT faces challenges in the tradeoff between accuracy and efficiency. Conventional implementations of GBDT need to scan all data instances to estimate the information gain of all the possible split points. Therefore, their computational complexities are proportional to the number of features and the number of instances. It is very time-consuming when handling big data. [42] proposes LightGBM with two novel techniques to solve performance issues by reducing the number of data instances and features, as follows:

- Gradient-based One-Side Sampling (GOSS): since instances with large gradients contribute more to information gain, when downsampling data instances, in order to retain the accuracy of information gain estimation, we keep those instances with large gradients and only randomly drop those instances with slight gradients.

- Exclusive Feature Bundling (EFB): in real applications, the feature space is relatively sparse, which allows us to design a nearly lossless approach to reduce the number of practical features.

4.1.3 CatBoost

Most gradient boosting implementations use decision trees as base predictors. However, many datasets include categorical features in practice, and it is not necessarily comparable with each other for the categorical feature having a discrete set of values. Then, in 2018, [26] proposed a new algorithm called CatBoost. It is the gradient boosting algorithm that processes categorical features in the training stage instead of using preprocessing time. The categorical features are converted to numbers at the preprocessing stage. Moreover, any combination of the categorical features will be considered as a new feature. CatBoost does

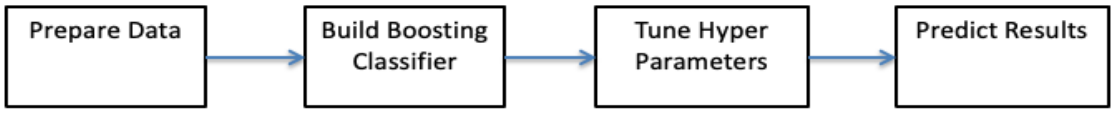


Figure 4.2: Workflow of the Anomaly detection using boosting method

not take combinations in the first split but combines all categorical features in the next split in the tree. The number of appearances of the category is an essential statistic in feature combinations as well. A new schema is introduced to calculate leaf values in selecting tree structure to reduce overfitting. [7] also highlights that CatBoost is sensitive to hyper-parameter settings, which creates discrepancies in training time performance.

4.2 Workflow

In this section, we describe the proposed workflow for anomaly detection. The workflow of the detection model is presented in Figure 4.2. Recall the description of the three datasets in BATADAL in Chapter 2.3. Since Dataset 1 does not have anomalies, Dataset 1 is not used in this model. Dataset 2 and Dataset 3 are used for training and testing. The description of each dataset can be found in Chapter 2.3.2. The training phase focuses on fine-tuning the hyper parameters, and the testing phase predicts the results. Next, an overview of workflow in Figure 4.2 is presented.

4.2.1 Data Preparation

The Dataset 2 and Dataset 3 raw data and their labels are the inputs for the models. 20% of the raw data is used for validation.

4.2.2 Anomaly Detection

XGBoost, LightGBM, and CatBoost are used to build the classifier. As the model's performance depends on the hyper parameters, the research uses grid search to determine the optimal values. Table 4.1 presents the most optimal hyper parameters for the three boosting models. K-fold cross-validation is applied, while k is set as 5. Then the raw data is split to 80% for the training dataset and 20% for the testing dataset in the data preparation stage in each run. The classifier is built using the training dataset, and the trained model is then used to predict the anomalies in the testing dataset.

Model	Hyper Parameters
XGBoost	max_depth=5, learning_rate=0.5
LightGBM	num_leaves=10, objective="binary", max_depth=5, learning_rate=0.5, max_bin=20, metric=["auc", "binary_logloss"]
CatBoost	eval_metric="AUC", depth=5, iterations=200, l2_leaf_reg=9, learning_rate=0.5

Table 4.1: Hyper Parameters for XGBoost, LighGBM and CatBoost

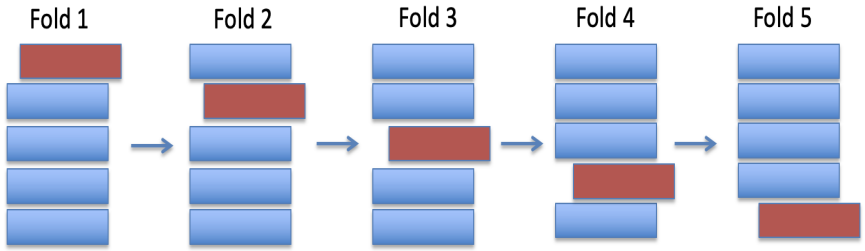


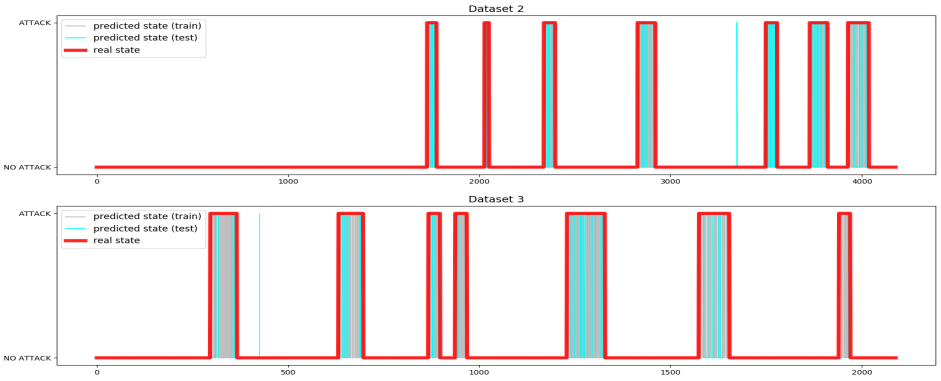
Figure 4.3: K-folds cross validation illustration. Training data is in blue while testing data is in red.

K-Folds Cross Validation

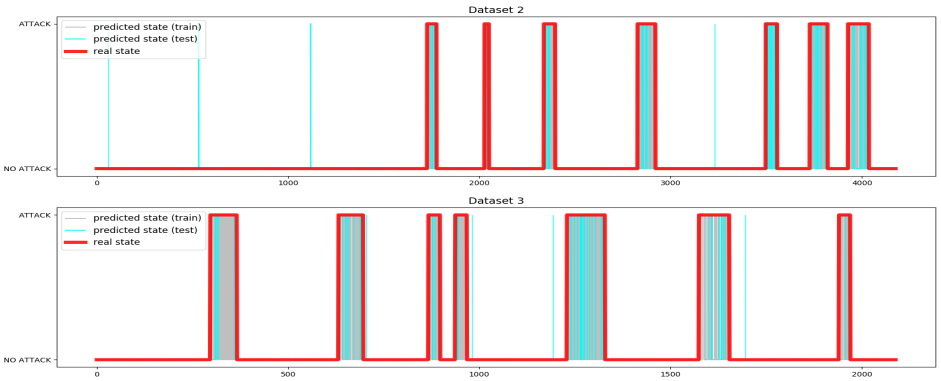
K-fold cross validation is introduced to evaluate the model performance [69]. It can shuffle a given dataset randomly, divide it into K number of folds, and then test each fold by using it as the testing dataset and the rest as the training dataset. Figure 4.3 illustrates the K-folds cross validation while K is set as 5.

4.3 Results

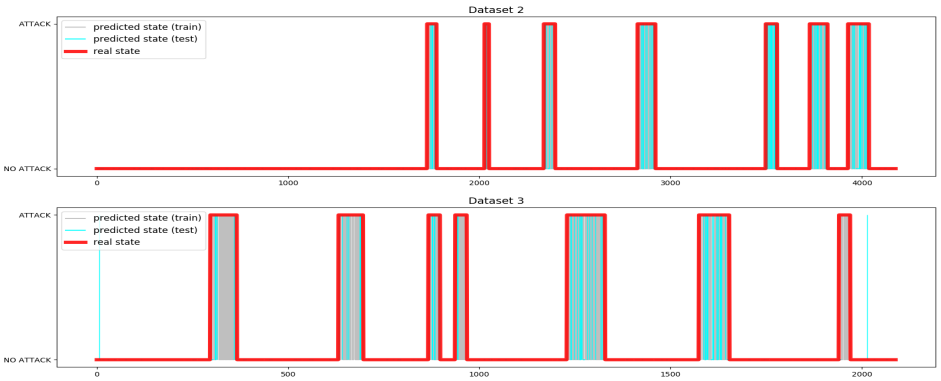
The model is built based on the workflow described in Section 4.2. We evaluate the results by using the performance metrics of recall, precision, F1 score, and accuracy. Table 4.2 presents the model's performance. As the attack and non-attack are not equally allocated, the table uses the weighted average for the performance metrics. After fine-tuning the hyperparameters, CatBoost gets more accurate results comparing with the other two methods. Meanwhile, the average time required to run the anomaly detection using boosting methods is recorded in Table 4.3. CatBoost needs the most time to finish the training and testing process. Figure 4.2 illustrates one of the test cases by performing the proposed three boosting methods.



(a) XGBoost



(b) LightGBM



(c) CatBoost

Figure 4.4: Result for anomaly detection using boosting methods. Red indicates the ground truth, i,.e, presence of anomaly, gray indicates the anomalies in the training process, and blue indicates predicted anomalous events for the testing dataset

	Dataset 2				Dataset 3			
Model	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall
XGBoost	0.944	0.937	0.946	0.944	0.895	0.880	0.904	0.895
LightGBM	0.935	0.928	0.935	0.935	0.897	0.886	0.899	0.897
CatBoost	0.952	0.947	0.955	0.952	0.897	0.884	0.903	0.897

Table 4.2: Result of anomaly detection using boosting methods based on accuracy, F1,precision and recall

	Dataset 2		Dataset 3	
Model	Training Time	Predict Time	Training Time	Predict Time
XGBoost	695.741	3.687	91.040	4.155
LightGBM	198.386	6.541	53.760	4.652
CatBoost	1495.243	12.315	1665.137	9.628

Table 4.3: Average time required to run anomaly detection using boosting methods. Unit is millisecond.

4.4 Conclusion

The proposed boosting models use XGBoost, LightGBM, and Catboost to detect cyber-attacks in the BATADAL dataset. The models are supervised machine learning and prediction-based methods. The XGBoost model gets the average performance metrics: accuracy of 0.9195, precision of 0.9250, recall of 0.9195, and F1 score of 0.9085; the LightGBM model gets the average performance metrics: accuracy of 0.9160, precision of 0.9170, recall of 0.9160, and F1 score of 0.9070; and the CatBoost model gets the average performance metrics: accuracy of 0.9245, precision of 0.9290, recall of 0.9245, and F1 score of 0.9155. The performance of CatBoost is slightly better than the performance of XGBoost and LightGBM based on the metrics.

Chapter 5

Anomaly Detection Application: Leak Detection in WDS

Leakage is one of the significant challenges in WDS. In Chapter 2, Chapter 3, and Chapter 4, the thesis introduces multiple anomaly detection algorithms. This chapter focus on applying anomaly detection methods to leak detection. The dataset used to evaluate the algorithm is from the Hanoi and Net1 network in leakDB, introduced in Chapter 2.3.2.

5.1 Workflow

In this section, we explain the workflow for leak detection on the water network Hanoi and Net1 in leakDB. The summary of leakDB has been described in Chapter 2.3.3. The workflow includes three phases, as shown in Figure 5.1. Data preparation focuses on providing required inputs for the training and testing phases. In the training and testing phase, boosting modules are used to find the optimal model and hyperparameters, and to make prediction given new data.

5.1.1 Data Preparation

The scenarios are divided into two groups based on whether they contain leakage. A new data file is generated by concentrating the scenario's demand, pressure, and flow sensors' time-series datasets. The new data frame for each scenario is presented in Figure 5.2.

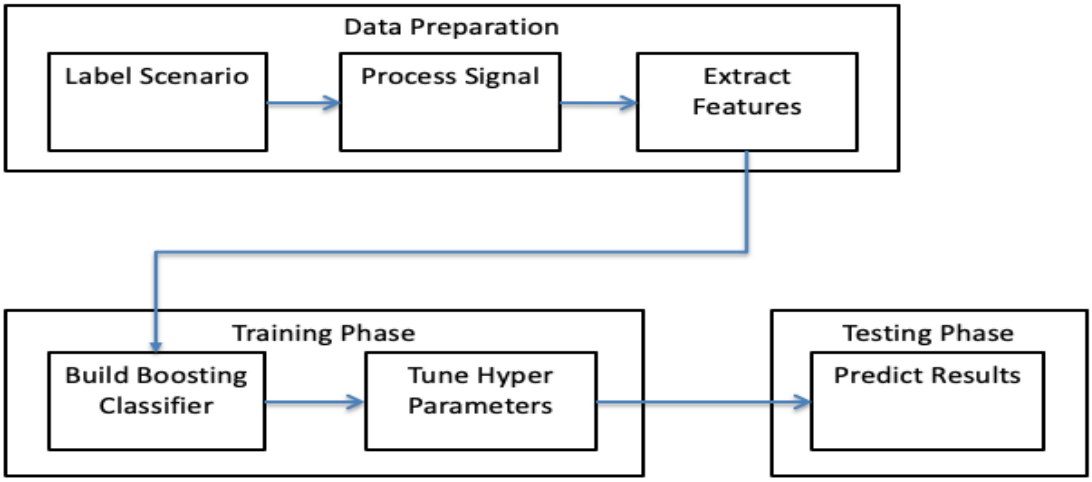


Figure 5.1: Workflow of the leak detection

	Timestamp	D_1	D_2	D_3	...	F_32	F_33	F_34
0	2017-01-01 00:00:00	-3330.0	154.8	169.2	...	-28.8	46.8	212.4
1	2017-01-01 00:30:00	-2944.8	126.0	147.6	...	-28.8	43.2	176.4
2	2017-01-01 01:00:00	-2574.0	108.0	126.0	...	-28.8	43.2	162.0
3	2017-01-01 01:30:00	-2368.8	108.0	108.0	...	-25.2	36.0	169.2
4	2017-01-01 02:00:00	-2149.2	100.8	108.0	...	-18.0	28.8	147.6

Figure 5.2: Scenario Data Frame. "D" represents demand, "P" represents pressure, and "F" represents flow.

5.1.2 Signal Processing

LeakDB produces the water demand, pressure, and flow data in 15 minutes resolution over 30 days for each scenario. To reduce the effects of human activities on the data, we process datasets in three steps: retrieve the data streams in night hours from each sensor node; downsample them by taking the daily night average values; then normalize them by scaling their values to the given range between zero and one. Sample is given in Figure 5.3(a) and 5.3(b) by plotting the raw and processed signals from Scenario-3 Demands Node 3, Pressure Node 3 and Flow Link 3 in Hanoi WDS.

5.1.3 Feature Extraction

Features can describe the characteristics of time series and reduce the Dimension. In this research, we extract mean, median, standard deviation (STD), kurtosis, and skew [28] [56] from the time series to represent the signal. Signal in Figure 5.3(b) can be described in Table 5.1. The feature extraction is applied to 98 signals in Hanoi WDS and 35 signals in Net1 WDS in all scenarios.

	Mean	Median	STD	Kurtosis	Skew
Demand	0.433474	0.674654	0.317684	0.435897	0.743429
Pressure	0.192034	0.232479	0.265256	0.302258	-0.746401
Flow	0.608928	0.601980	0.139184	0.744136	0.918281

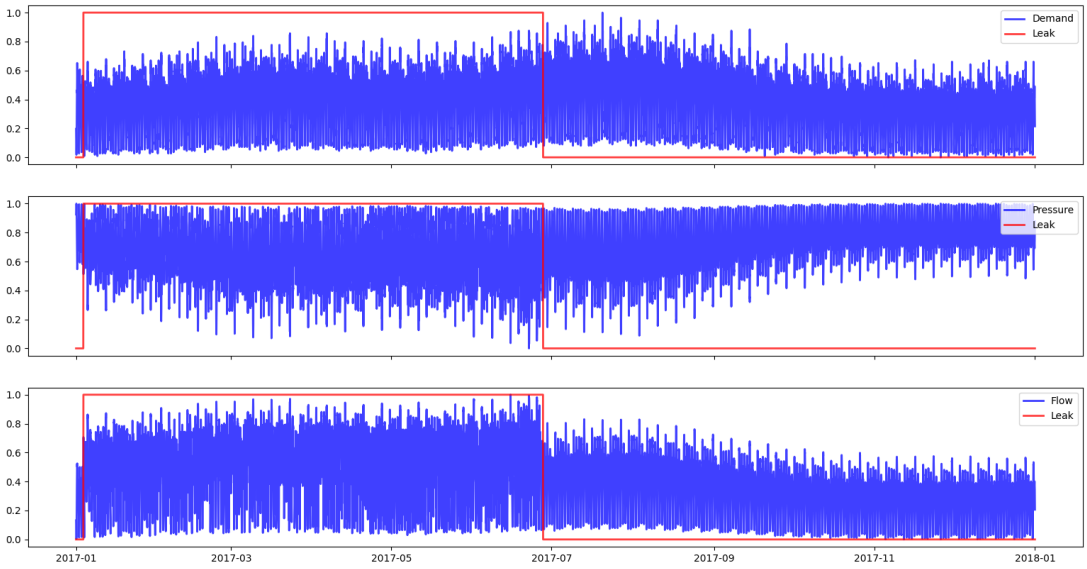
Table 5.1: Feature Representation of Demand, Pressure and Flow signals in Figure 5.3(b)

5.1.4 Anomaly Detection

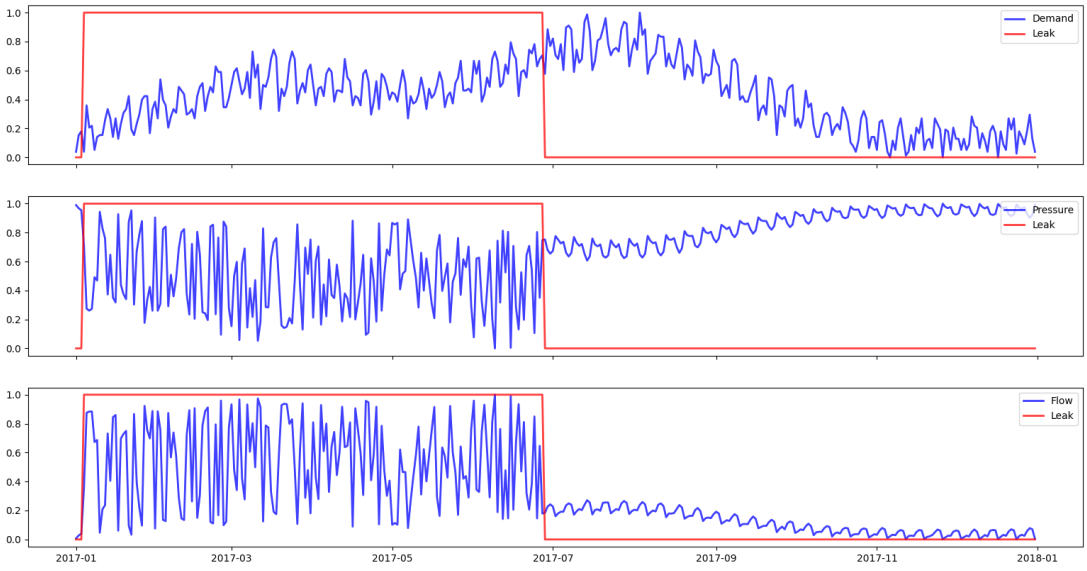
The XGBoost, LightGBM, and CatBoost methods are applied to the datasets of features. K-fold cross-validation with k=5 is applied. Then the input feature datasets are split to 80% for the training dataset and 20% for the testing dataset. Hyperparameters fine-tuning uses grid search following the steps in Chapter 4. The classifier is built using the training dataset, and the trained model is then used to predict the anomalies in the testing dataset.

5.2 Results

The results from three models are evaluated by performance metrics of recall, precision, F1 score, and accuracy. Table 5.2 and Table 5.3 present each model's performance. As the



(a) Raw Signal



(b) Night Average Signal

Figure 5.3: Preprocess the demand, pressure and flow signal. The x-axis is the time, and the y-axis is the measurements.

leak and non-leak are not equally allocated, the table uses the weighted average for the performance metrics. MNF 20% is the benchmarking algorithm described in Chapter 2.3.3.

Model	Accuracy	F1	Precision	Recall
MNF 20% [86]	/	0.565	/	0.399
XGBoost	0.925	0.926	0.929	0.925
LightGBM	0.945	0.946	0.950	0.945
CatBoost	0.940	0.941	0.944	0.940

Table 5.2: Result of leak detection in Hanoi WDS using boosting method based on accuracy, F1, precision and recall

Model	Accuracy	F1	Precision	Recall
XGBoost	0.980	0.980	0.981	0.980
LightGBM	0.985	0.985	0.986	0.985
CatBoost	0.975	0.975	0.976	0.975

Table 5.3: Result of leak detection in Net1 WDS using boosting method based on accuracy, F1, precision and recall

5.3 Conclusion

XGBoost, LightGBM, and Catboost anomaly detection models are applied to the Hanoi and Net1 network in LeakDB to classify leak and non-leak scenarios. The three models get the average performance metrics: accuracy of 0.9367, precision of 0.9410, recall of 0.9367, and F1 score of 0.9377 in Hanoi network; and accuracy of 0.9800, precision of 0.9810, recall of 0.9800, and F1 score of 0.9800 in Net1 network. The performance of LightGBM is slightly better than the performance of XGBoost and CatBoost based on the metrics.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This research works on semi-supervised anomaly detection and supervised anomaly detection. In semi-supervised anomaly detection, it proposes the matrix profile and autoencoder based ensemble method for anomaly detection. The model firstly uses the matrix profile to perform UTS anomaly detection on each sensor signal and secondly correlates the anomalies from multiple sensors to have MTS anomaly detection. Then, it integrates autoencoders to filter false alarms. The detection method is validated on the simulated cyber attack datasets, BATADAL. The matrix profile detects the anomalies with an average accuracy of 0.8175, precision of 0.4420, recall of 0.9705, and F1 score of 0.6060. After the autoencoder integration, the ensemble method achieves an accuracy of 0.9645, precision of 0.9385, recall of 0.8325, and F1 score of 0.8800. The localization is automated derived from the results of UTS detections.

In supervised anomaly detection, it proposes three boosting methods. They are XGBoost [16], LightGBM [42], and CatBoost [26]. The three boosting models split the BATADAL datasets into two parts: 20% for training and 80% for testing. The CatBoost model gets the best performance metrics among the three models. It gets the average performance metrics: accuracy of 0.9245, precision of 0.9290, recall of 0.9245, and F1 score of 0.9155.

The research applies anomaly detection models on simulated leakage datasets, called LeakDB. One thousand scenarios in the Hanoi water network in LeakDB are utilized in building the model and evaluating the model. Since the leak and non-leak labels are attached to the datasets, the three boosting-based anomaly detection models are applied. Instead of taking the time series directly, the five features, including mean, median, standard deviation, kurtosis, and skew, are selected to represent the time series. LightGBM gets slightly better performance metrics among the three models. It gets an accuracy of 0.945, precision of 0.950,

recall of 0.945, and F1 score of 0.946 in the Hanoi network; an accuracy of 0.985, precision of 0.986, recall of 0.985, and F1 score of 0.985 in the Net1 network.

6.2 Future Work

The matrix profile is a distance-based approach in time series anomaly detection. It calculates the euclidean distance (ED) in comparing the test datasets with labeled datasets. In this research, we used a list of window lengths instead of selecting one window length for matrix profile's calculation. The window length selection decides the time and resource taken for the model. It is time-consuming if the amount of time series data grows exponentially in the big data era. In the future, we are going to explore intelligent approaches for window selection.

The false alarms from boosting-based methods are from false-positive detections. As learned from the ensemble detection model using matrix profile and autoencoders, in the future, we can integrate boosting with other methods to have an initial step on detecting the period that anomalies are highly expected. In this way, we exclude the alarms raised by the non-likelihood anomaly time range.

Bibliography

[1] Sridhar Adepu, Venkata Reddy Palleti, Gyanendra Mishra, and Aditya Mathur. Investigation of cyber attacks on a water distribution system. In *International Conference on Applied Cryptography and Network Security*, pages 274–291. Springer, 2020.

[2] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

[3] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. Outlier detection. *The state of the art in intrusion prevention and detection*, pages 3–21, 2014.

[4] Hermine N Akouemo and Richard J Povinelli. Probabilistic anomaly detection in natural gas time series data. *International Journal of Forecasting*, 32(3):948–956, 2016.

[5] Mennatallah Amer and Markus Goldstein. Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer. In *Proc. of the 3rd RapidMiner Community Meeting and Conference (RCOMM 2012)*, pages 1–12, 2012.

[6] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.

[7] Andreea Anghel, Nikolaos Papandreou, Thomas Parnell, Alessandro De Palma, and Haralampos Pozidis. Benchmarking and optimization of gradient boosting decision tree algorithms. *arXiv preprint arXiv:1809.04559*, 2018.

[8] Karl Johan Åström. On the choice of sampling rates in parametric identification of time series. *Information Sciences*, 1(3):273–278, 1969.

[9] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

[10] Henri Begleiter and Lester Ingber. Uci machine learning repository: Eeg database data set, 1999.

[11] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. *arXiv preprint arXiv:2005.02359*, 2020.

[12] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *arXiv preprint arXiv:2002.04236*, 2020.

[13] Ronald Newbold Bracewell and Ronald N Bracewell. *The Fourier transform and its applications*, volume 31999. McGraw-Hill New York, 1986.

[14] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[15] Samprit Chatterjee and Martin Mächler. Robust regression: A weighted least squares approach. *Communications in Statistics-Theory and Methods*, 26(6):1381–1394, 1997.

[16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[17] David A Clifton, Peter R Bannister, and Lionel Tarassenko. A framework for novelty detection in jet engine vibration data. In *Key engineering materials*, volume 347, pages 305–310. Trans Tech Publ, 2007.

[18] John H Cochrane. Time series for macroeconomics and finance. *Manuscript, University of Chicago*, 15:16, 2005.

[19] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[20] Hoang Anh Dau and Eamonn Keogh. Matrix profile v: A generic technique to incorporate domain knowledge into motif discovery. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 125–134, 2017.

[21] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750–757, 2008.

[22] Sukhpreet Singh Dhaliwal, Abdullah-Al Nahid, and Robert Abbas. Effective intrusion detection system using xgboost. *Information*, 9(7):149, 2018.

[23] Peter Diggle. *Time Series: A Biostatistical Introduction*. Oxford University Press, 1990.

[24] RR Dighade, M Kadu, and AM Pande. Challenges in water loss management of water distribution systems in developing countries. *International Journal of Innovative Research in Science, Engineering and Technology*, 3(6):13838–13846, 2014.

[25] Meimei Ding and Hui Tian. Pca-based network traffic anomaly detection. *Tsinghua Science and Technology*, 21(5):500–509, 2016.

[26] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[27] Alireza Entezami, Hashem Shariatmadar, and Abbas Karamodin. Data-driven damage diagnosis under environmental and operational variability by novel statistical pattern recognition methods. *Structural Health Monitoring*, 18(5-6):1416–1443, 2019.

[28] Richard A Groeneveld and Glen Meeden. Measuring skewness and kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 33(4):391–399, 1984.

[29] John Hancock and Taghi M Khoshgoftaar. Medicare fraud detection using catboost. In *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 97–103. IEEE, 2020.

[30] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Overview of supervised learning. In *The elements of statistical learning*, pages 9–41. Springer, 2009.

[31] Ville Hautamaki, Ismo Karkkainen, and Pasi Franti. Outlier detection using k-nearest neighbour graph. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 430–433. IEEE, 2004.

[32] Paul Heckbert. Fourier transforms and the fast fourier transform (fft) algorithm. *Computer Graphics*, 2:15–463, 1995.

[33] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[34] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, 454(1971):903–995, 1998.

[35] Xiaohui Huang, Yunming Ye, Liyan Xiong, Raymond YK Lau, Nan Jiang, and Shaokai Wang. Time series k-means: A new k-means type smooth subspace clustering for time series data. *Information Sciences*, 367:1–13, 2016.

[36] Osama Hunaidi, Wing Chu, Alex Wang, and Wei Guan. Detecting leaks in plastic pipes. *Journal-American Water Works Association*, 92(2):82–94, 2000.

[37] DHS ICS-CERT. Us department of homeland security–industrial control systems-cyber emergency response team, year in review. 2014.

[38] DHS ICS-CERT. Us department of homeland security–industrial control systems-cyber emergency response team, year in review. 2015.

[39] DHS ICS-CERT. Us department of homeland security–industrial control systems-cyber emergency response team, year in review. 2016.

[40] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[41] Noy Kadosh, Alex Frid, and Mashor Housh. Detecting cyber-physical attacks in water distribution systems: One-class classifier approach. *Journal of Water Resources Planning and Management*, 146(8):04020060, 2020.

[42] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[43] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 151–162, 2001.

[44] Jihyun Kim, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. Long short term memory recurrent neural network classifier for intrusion detection. In *2016 International Conference on Platform Technology and Service (PlatCon)*, pages 1–5. IEEE, 2016.

[45] B Ravi Kiran, Dilip Mathew Thomas, and Ranjith Parakkal. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *Journal of Imaging*, 4(2):36, 2018.

[46] Istvan Kiss, Béla Genge, Piroska Haller, and Gheorghe Sebestyén. Data clustering-based anomaly detection in industrial control systems. In *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 275–281. IEEE, 2014.

[47] Katherine A Klise, Michael Bynum, Dylan Moriarty, and Regan Murray. A software framework for assessing the resilience of drinking water systems to disasters with an example earthquake case study. *Environmental modelling & software*, 95:420–431, 2017.

[48] Moshe Kravchik and Asaf Shabtai. Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca. *IEEE Transactions on Dependable and Secure Computing*, 2021.

[49] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering: a kernel approach. *Machine learning*, 74(1):1–22, 2009.

[50] Roland Kwitt and Ulrich Hofmann. Robust methods for unsupervised pca-based anomaly detection. *Proc. of IEEE/IST WorNshop on Monitoring, AttacN Detection and Mitigation*, pages 1–3, 2006.

[51] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22(1):949–961, 2019.

[52] Juan Li, Ying Wu, and Changgang Lu. Pipeline leak detection using the multiple signal classification-like method. *Journal of Hydroinformatics*, 22(5):1321–1337, 2020.

[53] Kun-Lun Li, Hou-Kuan Huang, Sheng-Feng Tian, and Wei Xu. Improving one-class svm for anomaly detection. In *Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE Cat. No. 03EX693)*, volume 5, pages 3077–3081. IEEE, 2003.

[54] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[55] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, volume 89, pages 89–94. Presses universitaires de Louvain, 2015.

[56] Kanti V Mardia. Measures of multivariate skewness and kurtosis with applications. *Biometrika*, 57(3):519–530, 1970.

[57] Larry W Mays. *Water distribution system handbook*. McGraw-Hill Education, 2000.

[58] ME Medhat, A Albdel-hafiez, MF Hassan, MA Ali, and Z Awaad. A review on applications of the wavelet transform techniques in spectral analysis. 2004.

[59] Elena-Adriana Minastireanu and Gabriela Mesnita. Light gbm machine learning algorithm to online click fraud detection. *J. Inform. Assur. Cybersecur*, 2019, 2019.

[60] Vladimiro Miranda, Adriana R. Garcez Castro, and Shigeaki Lima. Diagnosing faults in power transformers with autoassociative neural networks and mean shift. *IEEE Transactions on Power Delivery*, 27(3):1350–1357, 2012.

[61] Abdullah Mueen. Time series motif discovery: dimensions and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(2):152–159, 2014.

[62] Gerhard Münz, Sa Li, and Georg Carle. Traffic anomaly detection using k-means clustering. In *GI/ITG Workshop MMBnet*, pages 13–14, 2007.

[63] Henri J Nussbaumer. The fast fourier transform. In *Fast Fourier Transform and Convolution Algorithms*, pages 80–111. Springer, 1981.

[64] Ministry of Trade and Industry Department of Statistics. Retail sales index, (2014 = 100), at constant prices, (ssic 2010), quarterly, 2017.

[65] Avi Ostfeld, Elad Salomons, Lindell Ormsbee, James G Uber, Christopher M Bros, Paul Kalungi, Richard Burd, Boguslawa Zazula-Coetzee, Teddy Belrain, Doosun Kang, et al. Battle of the water calibration networks. *Journal of Water Resources Planning and Management*, 138(5):523–532, 2012.

[66] Ram Shankar Pathak. *The wavelet transform*, volume 4. Springer Science & Business Media, 2009.

[67] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[68] Pranav Rajpurkar, Awni Y Hannun, Masoumeh Haghpanahi, Codie Bourn, and Andrew Y Ng. Cardiologist-level arrhythmia detection with convolutional neural networks. *arXiv preprint arXiv:1707.01836*, 2017.

[69] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. *Encyclopedia of database systems*, 5:532–538, 2009.

[70] D Reynolds. Gaussian mixture models. encyclopedia of biometrics 827–832, 2015.

[71] Manel Rhif, Ali Ben Abbes, Imed Riadh Farah, Beatriz Martínez, and Yanfang Sang. Wavelet transform application for/in non-stationary time-series analysis: a review. *Applied Sciences*, 9(7):1345, 2019.

[72] Lewis A Rossman. The epanet programmer's toolkit for analysis of water distribution systems. In *WRPMD'99: Preparing for the 21st Century*, pages 1–10. 1999.

[73] Lewis A Rossman et al. Epanet users manual. 1994.

[74] Peter J Rousseeuw and Mia Hubert. Anomaly detection by robust statistics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(2):e1236, 2018.

[75] Krishnann Saravanan, Elraj Anusuya, Raghvendra Kumar, et al. Real-time water quality monitoring using internet of things in scada. *Environmental monitoring and assessment*, 190(9):1–16, 2018.

[76] Robert E Schapire. A brief introduction to boosting. In *Ijcai*, volume 99, pages 1401–1406. Citeseer, 1999.

[77] Robert E Schapire. The boosting approach to machine learning: An overview. *Nonlinear estimation and classification*, pages 149–171, 2003.

[78] Robert E Schapire and Yoav Freund. Boosting: Foundations and algorithms. *Kybernetes*, 2013.

[79] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING, 2003.

[80] M Sifuzzaman, M Rafiq Islam, and MZ Ali. Application of wavelet transform and its advantages compared to fourier transform. 2009.

[81] Jill Slay and Michael Miller. Lessons learned from the maroochy water breach. In *International conference on critical infrastructure protection*, pages 73–82. Springer, 2007.

[82] Riccardo Taormina and Stefano Galelli. Deep-learning approach to the detection and localization of cyber-physical attacks on water distribution systems. *Journal of Water Resources Planning and Management*, 144(10):04018065, 2018.

[83] Riccardo Taormina, Stefano Galelli, Nils Ole Tippenhauer, Elad Salomons, Avi Ostfeld, Demetrios G Eliades, Mohsen Aghashahi, Raanju Sundararajan, Mohsen Pourahmadi, M Katherine Banks, et al. Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks. *Journal of Water Resources Planning and Management*, 144(8):04018048, 2018.

[84] Sahar Torkamani and Volker Lohweg. Survey on time series motif discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(2):e1199, 2017.

[85] Vincent Vercruyssen, Wannes Meert, Gust Verbruggen, Koen Maes, Ruben Baumer, and Jesse Davis. Semi-supervised anomaly detection with an application to water analytics. In *2018 ieee international conference on data mining (icdm)*, volume 2018, pages 527–536. IEEE, 2018.

[86] Stelios G Vrachimis, Marios S Kyriakou, et al. Leakdb: A benchmark dataset for leakage diagnosis in water distribution networks. In *WDSA/CCWI Joint Conference Proceedings*, volume 1, 2018.

[87] Yung-Hung Wang, Chien-Hung Yeh, Hsu-Wen Vincent Young, Kun Hu, and Men-Tzung Lo. On the computational complexity of the empirical mode decomposition algorithm. *Physica A: Statistical Mechanics and its Applications*, 400:159–167, 2014.

[88] Dazhi Yang and Stefano Alessandrini. An ultra-fast way of searching weather analogs for renewable energy forecasting. *Solar Energy*, 185:255–261, 2019.

[89] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 1317–1322. Ieee, 2016.

[90] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

[91] Victor Zarnowitz and Ataman Ozyildirim. Time series decomposition and measurement of business cycles, trends and growth cycles. *Journal of Monetary Economics*, 53(7):1717–1739, 2006.

[92] Dengsheng Zhang. Wavelet transform. In *Fundamentals of Image Data Mining*, pages 35–44. Springer, 2019.

[93] Ming Zhang, Boyi Xu, and Jie Gong. An anomaly detection model based on one-class svm to detect network intrusions. In *2015 11th International conference on mobile ad-hoc and sensor networks (MSN)*, pages 102–107. IEEE, 2015.

[94] Zijun Zhang and Andrew Kusiak. Monitoring wind turbine vibration based on scada data. *Journal of Solar Energy Engineering*, 134(2), 2012.

[95] Yan Zhu, Chin-Chia Michael Yeh, Zachary Zimmerman, Kaveh Kamgar, and Eamonn Keogh. Matrix profile xi: Scrimp++: time series motif discovery at interactive speeds. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 837–846. IEEE, 2018.

[96] Yan Zhu, Zachary Zimmerman, Nader Shakibay Senobari, Chin-Chia Michael Yeh, Gareth Funning, Abdullah Mueen, Philip Brisk, and Eamonn Keogh. Matrix profile ii: Exploiting a novel algorithm and gpus to break the one hundred million barrier for time series motifs and joins. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 739–748. IEEE, 2016.