Event Processing in Wireless Sensor Networks

Somsri Jarupadung

Submitted for the Degree of Master of Philosophy
University of Surrey

Institute for Communication Systems
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, UK

December 2014

© Somsri Jarupadung 2014
Abstract

Event detection has been studied and researched for many years and it has been applied in real world applications with the aim of characterising a situation in the real world. In order to capture a situation, Wireless Sensor Networks (WSNs) are deployed and sensor nodes are used to sense the entities of interest for the real world application; sensing the environment results in the production of a large and often continuous production of raw data. In this context, event detection is used in order to extract the most relevant and useful information from this large set of data. The constraints of nodes have to be taken into account such as energy, computation, and memory.

The environment is observed from a program that is hosted on a sensor node. Machine learning and data mining techniques are embedded in the program to learn from the environment and detect events. A collaborative sensing is a technology to process an event from distrusted nodes which can enhance an accuracy result that can be fault or event.

This research studied processing sensor data to detect events using multiple sensor nodes. A model and/or rules are defined in order to detect an outlier from data matching between sensor data and the model and/or rules. An outlier is analysed and processed to detect an event.

The main contributions of this work have been on collaborative sensing in different sensors including clustering analysis for data labelling, classification analysis in order to process an outlier for an event detection.

Key words: Event Detection, Event Processing, Wireless Sensor Network, Machine Learning, Collaborative Sensing

Email: s.jarupadung@surrey.ac.uk
Contents

Abstract .......................................................................................................................... ii

Contents ..................................................................................................................... iii

List of Figures ............................................................................................................ v

List of Tables .............................................................................................................. vii

1 Introduction .............................................................................................................. 1

1.1 Motivations and Objectives ................................................................................. 3

1.2 Problem Definition ............................................................................................... 6

1.3 Challenges ............................................................................................................. 7

1.4 Methodology ......................................................................................................... 8

1.5 Main Contributions ............................................................................................... 9

1.6 Thesis Outline ..................................................................................................... 9

2 Background ............................................................................................................. 10

2.1 Sensor Data Processing ....................................................................................... 10

2.2 Wireless Sensor Networks .................................................................................. 11

2.3 Event Detection ................................................................................................... 13

2.3.1 Predefined Patterns ....................................................................................... 14

2.3.2 Automated Methods ...................................................................................... 15

2.4 Collaborative Event Processing ......................................................................... 23

2.4.1 Knowledge Fusion and Aggregation ............................................................. 23

2.4.2 Distributed/Consolidated Sensing ................................................................. 26

2.4.3 Cluster Structuring ....................................................................................... 27

2.5 Discussion ........................................................................................................... 29

3 Data Processing for Event Detection ..................................................................... 30

3.1 Experiment Setup ............................................................................................... 30

3.2 Sensor Data Pre-Processing ............................................................................... 31

3.3 Cluster Data Processing ..................................................................................... 39

3.4 Summary and Discussion ................................................................................... 42
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Event Processing</td>
<td>43</td>
</tr>
<tr>
<td>4.1</td>
<td>Outlier Processing</td>
<td>43</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Classification Analysis</td>
<td>45</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Outlier Analysis</td>
<td>54</td>
</tr>
<tr>
<td>4.3</td>
<td>Summary and Discussion</td>
<td>56</td>
</tr>
<tr>
<td>5</td>
<td>Summary and Conclusion</td>
<td>57</td>
</tr>
<tr>
<td>5.1</td>
<td>Summary</td>
<td>57</td>
</tr>
<tr>
<td>5.2</td>
<td>Contributions</td>
<td>57</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Clustering Analysis</td>
<td>58</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Classification Analysis</td>
<td>58</td>
</tr>
<tr>
<td>5.3</td>
<td>Discussion and Future Work</td>
<td>59</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Discussion</td>
<td>59</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Future Work</td>
<td>59</td>
</tr>
<tr>
<td>5.4</td>
<td>Conclusion</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>62</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1.1: MTM-XM 1000 sensor node ......................................................... 2
Figure 1.2: Processing units in sensor node ..................................................... 2
Figure 2.1: Taxonomy of event detection solutions ......................................... 14
Figure 2.2: Collaborative Event Processing Taxonomy .................................... 24
Figure 2.3: Knowledge fusion and aggregation ................................................. 24
Figure 2.4: Distributed/consolidated sensing .................................................. 26
Figure 2.5: Cluster structuring ................................................................... 28
Figure 3.1: The meeting room layout for the experiment ................................. 31
Figure 3.2: The training dataset processing structure ......................................... 32
Figure 3.3: The process to create training datasets and clustering performance evaluation ............................................................... 34
Figure 3.4: Discrete wavelet transform’s block diagram ................................... 34
Figure 3.5: Rules for data labelling ................................................................. 36
Figure 3.6: The completeness score and homogeneous score of the cluster in varied window size of streaming sensor data ......................................................... 38
Figure 3.7: The comparison of data labelling between performing by statistical rules and k-means clustering ............................................................... 39
Figure 3.8: The processes of training data preparation ....................................... 41
Figure 4.1: Event detection processes ............................................................ 44
Figure 4.2: Event processing model ............................................................... 45
Figure 4.3: Classification processes .............................................................. 46
Figure 4.4: The output graph from each step in order to adjust data range for statistical values which are used for rules creation ......................................... 49
List of Figures

Figure 4.5: Rules for data labelling using adjusted dataset.......................... 50
Figure 4.6: Classification accuracy using over-peaked data removal and
manual labelled of test dataset and rules...................................................... 52
Figure 4.7: Computation time using over-peaked data removal and manual
labelled of test dataset and rules................................................................. 53
Figure 4.8: Online processes of outlier detection ....................................... 55
List of Tables

Table 2.1: Event processing techniques ................................................................. 22
Table 2.2: The performance comparisons between event detection technologies in WSN .................................................................................................................. 23
Table 3.1: Outlier detection comparison between different DWT functions based on the experiment that is completed in this research ............................................................... 35
Table 3.2: Cluster performance (training dataset) varied by window size and statistic values .................................................................................................................................. 37
Table 4.1: Classification accuracy using divided data of a training and testing set causing an over-fitting problem ................................................................................................. 46
Table 4.2: Classification accuracy using over-peaked data removal and manual labelled of test dataset and rules ................................................................................................. 52
Table 4.3: Computation time using over-peaked data removal and manual labelled of test dataset and rules ................................................................................................. 53
Chapter 1. Introduction

In a digital world, data can be retrieved from any devices at any time. A sensor is a digital data source which aims to process and extract information for human understanding in specific applications. Many systems utilise sensor data from its network (Wireless Sensor Networks: WSNs) to monitor the physical world in order to detect an occurrence in an area of interest. Occurrences can be an event abstracted in sensor data to inform users in real-time. In this sense, event processing moves beyond data retrieval to interpret a large volume of data into meaningful data.

A sensor device produces digital data from an environment observation such as humidity, sound and light which can be wireless sensor nodes, smart phones and GPSs. A node (Figure 1.1); e.g., the MTM-XM 1000, senses an environment from its sensors which consists of two light sensors, a temperature and humidity sensor. Sensors from such nodes sense the physical environment (e.g., light and humidity) as analogue signals. The processor, which is not high power (MSP430F2618), processes the sensing data and converts the data from analogue to digital (i.e. bits) using an ADC (Analogue to Digital Converter) periodically. The data is processed according to the instructions of the application installed in the memory that has a capacity limitation. The resulting data is processed to find the semantic or pattern within that stream. The communication between devices within the sensor node range is supported by the radio module (Figure 1.2). The power source for wireless sensor node comes from batteries. A microprocessor in the node can process the data, however, the energy consumption of the in-node computation is significantly less than the power consumption of the communication (considering the trade-off between communication and in-network processing of continuous data). The power consumption of the node is proportional to the distance it needs to communicate.

A node provides a large amount of data in the form of a bit sequence streaming by time (time series) that is an environment is captured into numerical value
continuously over period of time. The database in a node cannot support such data. In this case a database stores limited significant historical data and the former obsolete data is over written. The data for communication between nodes is usually only signature/important data in order to prolong the node’s lifetime since transmission time affects the energy usage.

Outlier analysis in sensor nodes aims to categorise between a normal situation and an occurrence. However, abnormal data may not only be the occurrence but also noise and errors which occur due to the resource limitation in the sensor node. In order to enhance the detection performance, data handling in a node or collaboration between nodes can lead to generate lower false detection and thus higher accuracy. The further processes find a semantic of a phenomenon to detect an event, which is then classified as a pattern of infrequent or abnormal occurrences in the area of interest [1], as the application requires.

Many applications utilise sensor data to observe an environmental phenomenon which is enabled by WSNs such as agricultural management, natural disaster alert systems, health monitoring and traffic flow control in order to inform/alert a user/actuator or perform a primary action when an event has occurred. The nodes can be installed anywhere for event processing to observe unusual
behaviour at any time and any place even in inaccessible locations. Some applications need to be detected and solved immediately which humans cannot do through their perceptions, moreover, certain events in the environment need to be observed and detected in real-time. For example, monitoring structural foundations, which cannot be accessed or measured by people directly, informs relevant engineers to check and maintain the foundation when abnormalities occur. Flood and tsunami detection can directly contribute to saving people’s lives if they have been informed before it happens. Forest fire detection is useful for the environment and animal life if fires can be detected before they spread into wider areas. Health care is an application where WSNs can detect events or abnormal situations from human. It can be used for monitoring the well-being of patients by equipping them with sensors to detect heart rate, blood pressure, movement, and so on. If a patient’s health is found to be in a high risk situation, the trigger will be sent to the doctors or nurses immediately to request help. Pollution monitoring enables people/subscribed users who want to know the pollution statistics in an area of interest to make decisions such as avoiding an area or wearing a filter mask. For water pollution, when the chemical or pollute in the water is higher than the safety level, people who consume the resources in that area should know to avoid using them and recovering processes must be executed. Moreover, they can be applied to monitoring and controlling systems in many applications, for example, agriculture in which observations of an environment such as temperature, light and humidity can be controlled to an appropriate situation for crops.

Observed data is most likely to come from a variety of sensor types which produce the huge amount of sensor data and only relevant information is sent to the user. The technologies which are applied in an event detection application have to support a huge amount of data and lead to decision-making such as machine learning, data mining and statistical methods.

1.1 Motivations and Objectives

Event detection is widely applied in various applications and performs with suitable technologies in its environment, application and user requirements.
Statistic methods, data mining and machine learning are the science directly
related to this area from the nature of data and applications.

WSNs in event detection are applied in a distributed system so collaborative
processing is considered in order to enhance the detection performance,
accuracy and reliability since it does not process and making a decision from a
node. Each node needs to exchange data with adjacent nodes/gateway node to
describe an occurrence. The resource limitation, especially power, is a major
issue for this type of system as power consumption for data transmission is very
high. Prolonging the nodes lifetime is important and much research in this area
is being pursued.

Sensor data reading is varied by space and time. The correlation between them
in the area of data interpretation can have a large impact; for example sound
levels decay over time and distance, i.e. the more distant sound sensing nodes
from the source measure a lower noise level. In terms of source movement, this
may be another case of sensor data interpretation which concerns spatial and
temporal data; an example for this case would be the noise from fast moving
emergency vehicles. The observed sound pattern information may give
important information about the traffic situation.

A streaming sensor data contains much more information than the simple static
levels obtained from a sensor. Time series data allows us to derive semantic
information/knowledge which then can be made available to users for further
analysis. This data normally has its own pattern in general situations but
outliers may appear on the signal. In order to detect an outlier, the “normal”
behaviour needs to be defined before the processing starts. WSN literature
covers two approaches for event detection:

**A) Outlier analysis** consists of noise, anomaly and event. Noise can be an error
from a sensor reading or communication but there is no pattern in that signal.
The model of the outlier is including an anomaly and an event. An event is a
known pattern with semantic data, however anomalous data has not been
defined as semantic. The technologies for outlier processing in WSNs data have
been proposed and developed for years to support sensor node constraints and
improve performances.
**B) Threshold based** event detection techniques have a constant value which separates between outliers and a normal situation. It focuses on defining a value that leads to faults from distinguish noises and occurrences. The technique is straightforward and simple so it suits for resource constraint devices like wireless sensor nodes, but there is a high possibility for false detection when this technique is implemented. However, it is a fundamental technology that can be adapted/assembled with other technologies.

There are a number of techniques to define (and redefine) the thresholds or “normal” patterns for the aforementioned two approaches.

**Machine learning and data mining** techniques are based on learning from a historical data/dataset to create a model in a particular environment. The statistic methods are usually embedded into this category in order to process the historical data. The model makes a more robust system to define a normal situation and event.

Event detection in WSNs has special characteristics which need to be considered from sensor reading data and communications as mentioned before. Improved detection performance reduces fault alarm rate in order to enhance the system reliability and accuracy. Since an erroneous data can be occurred from any processes, data processing and analysis in a sensor and a gateway node needs to support this system. Response time is a factor in some applications, however, a node cannot be in an active mode all of the time due to the power source limitations. The processing with large amounts of sensor data with low delay is an issue to be considered.

This work has the goal to improve the performance of event detection by analysing data streams. The approach aims to detect an event from observed data which generally has anomalous/outlier data in order to perform further analysis such as event processing.

This research has six main steps;

- Provide sensor data from different sensors in order to analyse them
- Pre-process data such as windowing, noise and data size reduction (discrete wavelet transform), over-peaked data removal
- Classify data and label it
- Create rules for data labelling, outlier detection and event processing
- Detect an outlier
- Analyse the outlier for event detection

1.2 Problem Definition

This thesis tackles the problem in event processing in WSNs focussing on *how can knowledge be extracted reliably from sensor data that may be noisy, contain anomalies and actual events?*

Data stream in a wireless sensor node is unreliable and may contain noise, errors and occurrences so data filtering for noise and errors is necessary to process only significant data. Since a node has resource constraints – energy, memory and computation, this can lead to inconsistency and unreliability of captured data. These limitations effect fault rates in a system especially when the power is very low which can be the cause of error in data generation [2]. Moreover, the erroneous data can also occur if a node has malfunctioned. Deploying WSNs for event detection has been concerned with understanding the nature of noise, errors from communication and sensor reading and incomplete received data in distributed systems in addition to resource constraints considerations.

As the resource constraint of the nodes, the techniques for event processing have to be chosen wisely such as low complexity and gives an acceptable outcome. Threshold defining for outlier detection is simple and suitable for restricted environment. However, it cannot be employed because of high false positive and the threshold can be different for different area. Learning the behaviour of an environment for model creation in order to discover an abnormal situation is possible by using several methods such as statistical analysis, machine learning and data mining techniques. To create a model, an observer or training dataset needs to be realistic.

WSNs in event detection is a distributed system where collaborative sensing has an important role in order to support the detection performance. The network structure and where the decision is made can be considered for collaborative processing because the processing is from more than a node/sensor. It is including knowledge fusion and aggregation, distributed/consolidated sensing...
and cluster structuring which is described in Chapter 2, section 2.4 Collaborative Event Processing. This work considers distributed/consolidated sensing (for the future work) which is in section 2.4.2 because the experiment is setup from a small area so the network is small and signals are propagated to make the processing complete in another sensor node.

Many techniques for event detection have been proposed for the constrained device of wireless sensor nodes. Collaborative sensing supports the nature of a distributed system to enhance the performances where machine learning is extensively studied. The drawback of these techniques mentioned before is static models where the patterns are created offline from selected dataset/historical data which needs to be chosen carefully.

1.3 Challenges

A sensor node has characteristics in resource constraints such as energy, computation, and memory, which provides challenges for applying the system in real world applications including hardware limitations, heterogeneous data sources, mobility, scalability and location dependence.

*Hardware limitations:* sensor nodes cannot be recharged with power and the nodes cannot be changed after installation so they cannot run the same algorithms as unlimited power source devices, which have no power source limitation. Energy aware event detection algorithms are very important to prolong nodes lifetime. Sensor devices have a limitation of computation because of the size and the complexity of board and chips. From this point, they cannot run complex instructions. If the workload is high, the response time would be reduced and the delay is increased. Moreover, there is a limitation of memory in the node so it cannot contain a huge sensor database and program so the processing in a node should be simple with less complexity. It has to be completed in small instructions and fast computations which leads to a fast response time with acceptable accuracy and fault.

*Scalability:* a stream of sensor data from a node is normally very large. Event processing in a node has to deal with this data. Moreover, the data is not from a single node, it comes from multiple nodes in a wide area. The event detection
algorithms have to manage and filter the right data to get an effective computation. If the event processing does not consider the scale of data, the traffic or transmission cost is very expensive and that can lead to other problems such as energy consumption from transmission cost and response time from traffic.

**Location dependence:** the pre-defined value or threshold to detect an event in different locations is not the same. Since different locations have different sensitive values in the same event, the location has to be considered in threshold setting. For example, fire detection in the kitchen and bedroom has a different value. The smoke detection in the bedroom should be more sensitive than the kitchen. In flood detection, there are sensitive areas, basin areas or low landscapes which cannot apply the same rules as high landscapes because of the flow and speed of water.

The technologies that make use of WSNs in event detection have been concerned with the node limitations and nature of its processing as mention before. However, this thesis is also concerned with the ability and detection performance for event detection. The aim of this thesis is to overcome the above challenges of event processing under these varying conditions.

### 1.4 Methodology

This research is applied to sensor data collected from the Smart Campus test-bed at the University of Surrey. This sensor data from nodes is processed to detect events such as a meeting in a room or a presentation in a room. Each node has sensors which captures temperature, light level, noise level, power usage and passive infrared with time stamp. The sampling frequency is 10 seconds. The experiment of this research applies sensor data in June for 4 working days to create a model for outlier detection. Data reduction and noise reduction is performed in a pre-processing step for overlapping sliding window data. The data is grouped into three groups using a clustering method and labelling the window data by statistical rules. A classifier model is created from this data. The model and statistical rules are utilised to process and analyse an event. This experiment utilises the model to detect a meeting in a meeting room.
by performing three different sensors including temperature, light level and noise level from four nodes.

1.5 Main Contributions

The main contributions of this research are listed below

- Developing a semi-automated clustering mechanism for labelling sensor data which is used for training dataset to create a classifier.
- Developing an outlier detection using a classification model and rules for an event analysis.

1.6 Thesis Outline

The remainder of this report is organised as follows.

Chapter 2 describes the associated background and state of the art with this thesis which include sensory data processing, wireless sensor networks, event detection and collaborative event processing. The event detection technologies are categorised into two groups; namely predefined patterns and automatic methods. The collaborative event processing is described in order to enhance the reliability which includes knowledge fusion and aggregation, distributed/consolidated sensing and cluster structuring.

Chapter 3 provides event processing which mainly is an offline process to create a classifier for outlier detection. It is started from data preparation to create the model and then to separate that data into groups and label them for training dataset. The rules is created to label the data. This chapter considers only single sensor and single node.

Chapter 4 provides event processing which process data from different sensors. A problem from the previous chapter is solved namely over-fitting. A model is proposed from pre-defined rules which considers data that is not in a normal curve and not in a normal distribution for data labelling. An online process is performed in order to detect an outlier in real environments using adjusted rules for different sensor data.

Chapter 5 describes a concise summary leading to conclusions and future work.
Chapter 2. Background

This chapter provides the related technologies to event detection in WSNs. These include sensory data processing, event detection and collaborative event detection which can improve the performances of the system.

2.1 Sensor Data Processing

Sensor data is generated from any sensor devices such as a node, smart phone or smart car which is produced from different manufacturers. However, they have the same significant components – microcontroller, radio module, memory, sensor and power source which are mentioned in the first chapter. They are the factors to design and deploy the system because of the resource constraints. The smart nodes are usually run under a supported operating system (OS) which is smaller than other OSs in other platforms and provides only nontrivial functions and friendly power consumption. Sensor data is transmitted to other nodes via radio transceiver in the form of radio waves and is then converted back to data bits at a sink.

Sending and receiving data in sensor node is a significant task to process an event in a network via transceiver module. The operational states consist of transmission, receiving, idling and sleep. The demand of power in transmission and receiving states is very high compared with other states. The idling is ready to receive data but some functions in the hardware are switched off to save a small amount of energy. However, the sleep state consumes less energy than idle since the significant parts of the transceiver are switched off but it needs a recovery time and start up energy to leave this state so it cannot receive data immediately as requested.

A wireless sensor node cannot be recharged after installed so all energy consumption factors are considered and taken into account even though ambient conditions can provide other sources of power by its design such as
solar cells, temperature gradients and vibrations. The compiler/interpreter and number of instructions are also considered but their effect is minimal. The microprocessor modes are significant but considered not to be in a full operational mode all the time. Sleeping mode can save energy however there is an overhead from turning into sleep/active state which consumes power without processing. The power consumption in a radio module and microprocessor is incomparable but both are important. The transceiver requires much more energy than the microprocessor and other modules so less communication is less power consumption which leads to in-network processing technologies.

Sensor data is processed in order to analyse a semantic in the data [52]. The data can be analysed in-node or integrated with other devices such as WSN and mobile cloud computing [3]. Single node processing is a primary analysis to detect an occurrence of an event however faults from sensor reading, noise and communication and so on lead to the lack of reliability. Collaborative processing is superior to single node detection in that the processed data is not from only a single source, therefore false detection from noise and sensor reading is reduced.

2.2 Wireless Sensor Networks

A sensor node normally senses an environment in order to measure something in that area and connects to other nodes by its nature. Data is mostly routed to a base station which has a specific target node, not arbitrary communication. Since all nodes have the same target, the traffic is unbalanced. However, communication characteristics take into account the traffic and performance in a network. Data communication is categorised into two groups, single hop and multi-hop. Single hop communication sends data directly from source to sink. The capability of this method depends on received signal strength indication (RSSI). The single hop transmission is simple to implement however the communication distance has a higher possibility be larger than multi-hop communication therefore more energy is consumed. The communication distance is an important factor for an energy consumption. Multi-hop transmission operates by sensor nodes via intermediate device(s) which can reduce the cost of transmission.
The communication between nodes is highly influenced by network protocol. The data transmission can be based on different designs such as time-based, event-based, query-based and hybrid.

- **Time-based:** the communication between sources and sinks is performed in periodic time scales. All sensors send data periodically which is suitable for applications that require periodic monitoring.

- **Event-based:** patterns of occurrences are defined in order to detect an event. Event-driven detects a changing environment and sends significant data when drastic changes are detected.

- **Query-based:** a responding signal is sent to a base station/another node when a query is processed and asks for the information of its environment.

- **Hybrid:** the transmission data is sent to other nodes by considering more than a key based design.

Event processing in WSN can be performed at a node, in-network and/or gateway node which receives data from a data generator – wireless sensor node. A gateway node has a higher capabilities and resources than a wireless sensor node so many applications perform full performance processing at this level. The gateway node is in the middle between user level and hardware level, and it is a connection between higher and lower level of sensor nodes from anywhere/any devices to interoperate heterogeneous devices in different formats into the same standard which a user can query and retrieve a knowledge from it. Ganz et al. [4] propose the middleware to manage the heterogeneous WSNs. There are three layers, connectivity, information processing, and service layer. Each layer is connected to knowledge base and control, and management module. Connectivity layer deals with the heterogeneous sensor hardware with different protocol and format for the node registration to get semantic description of each node, status and capabilities. The second layer contains algorithms and mechanisms to discover different patterns and/or events.

Users receive event signals/information which is a knowledge they want to retrieve from the system and response it via applications and/or services. This knowledge is shown and/or it is in the knowledge base for query. Boyaci et al. [5] develop the event-driven Sense Everything, Control Everything (SECE)
framework which is able to create nearly natural language for rules which is easy to understand for human. It is the integrated system with the Internet, cellular and sensor networks. An environment is monitored and controlled by the rules.

2.3 Event Detection

Event detection is one of the important data analyses in WSNs. An event is a member of outliers where it is useful to identify an abnormal situation in order to perform an action/decision-making when an event is detected. In sensor nodes, the power source is mainly used for communication tasks whereas, computation consumes comparatively less resources. Therefore, power consumption from communication is the main concern.

Identifying an event is important as it can cause the detection to be false or unreliable. The detection performances such as detection accuracy and computation time can be related from training dataset selection that is provided for detection model creation in order to generate detection models, rules or thresholds. The training dataset from the observed environment should be realistic and applied at the same area to support the detection results.

Semantic of the data results from observing phenomena while the sensor is capturing the environment. A pattern matching between capturing data and models is processed in order to interpret such data. Pattern matching algorithms and the models are an indicator of detection performance in reliability, accuracy and minimal faults. Moreover, the intelligent system leads to categorise other outliers such as noise and anomalous data and then perform other processes for a dynamic event detection system.

The event detection techniques have been proposed as taxonomy in Figure 2.1 which are categorised into two groups – predefined patterns and automatic methods. These techniques are considered to overcome the node limitations and detection performances. Predefined patterns are based on a threshold which is a user dependence whilst automatic methods are based on learning techniques from an environment data.
2.3.1 Predefined Patterns

Patterns are predefined by an expert user who knows the nature of the observed environment which plays a pivotal role to determine features and thresholds in each attribute in the area of interest. The processing complexity is very low since the process is a comparison between observed values and a threshold. There are benefits products of low processing such as low response time and less power consumption; unfortunately it does not support false positives which leads to unreliability and a lack of accuracy. An error, noise and occurrence cannot be distinguished since sensor data can contain all of them but the threshold is a value that has a function to categorise sensor data into two groups only, lower or equal and higher or equal than the threshold, it cannot recognise error and noise. There are techniques to take advantage of this domain and reduce these drawbacks. Gu et al. [6] propose multi-level detection by processing into four levels - sensor, node, group, and base level. In the lowest level, the signal is analysed using a threshold in each sensor type. The results are sent to the group level for further processing. Vu et al. [7] show composite event detection based on thresholds. Sub-events from different sensor nodes are analysed with the same or near time period. Xue et al. [8] propose five shapes to identify pattern of occurrences including horizon, slope, oscillation, jump and spike with predefined parameters from an expert user. A memory management
is provided using a two-level compression scheme in order to store a historical sensor data for matching processes. S.-J. Yim and Y.-H. Choi [9] propose two thresholds for filtering faults and noise in order to increase an accuracy and reduce faulty alarms from sensor data. With a fixed threshold, when the environment changes, the system can easily obtain a faulty detection, so drawbacks of these techniques are user dependency by threshold defining and a lack of automatic system.

The techniques in this category are user dependent which means a low of an automatic system, therefore the false rate might be increased. Moreover, the system is not tolerant from a user-predefined threshold since an environment can be changed from any factors and there are different conditions in space and time. The automatic system can support these problems and be suitable in real world systems.

2.3.2 Automated Methods

Models/patterns are created automatically from a historical data in real situations using supported technologies such as machine learning and data mining techniques by various learning methods. A training dataset or historical data from an observed environment is provided for the learning processes. The learning methods perform a model in order to categorise outliers and normal situations which are applied in matching processes for a data comparison between model and sensor data. The automatic event processing can be performed in large scale and complex environments because the learning processes create a model and analyse data with multiple input variables in a real environment. Moreover, it can be applied for the predicted system.

The algorithms in this category mostly have a learning output in the form of model, graph or tree for classifying data features. This section describes two main techniques for automatic event processing namely classification and clustering.

Classification is a supervised technique which needs to learn from a training dataset to generate a model before applying it in the system. The model/pattern is utilised to classify normal situations and abnormal phenomena for the further
event processing. Classification approaches have been examined in many works which are described into five categories in this document; complex models, tree structure, fuzzy logic and association rules, pattern based and hybrid. Clustering is an unsupervised technique which can group data into groups without learning from historical database. It requires more resources than a classification technique so there are not too many works have tried to focus on this technique in the constraint resources from the literature. This document describes two techniques namely adapted k-mean and distance metric.

2.3.2.1 Complex Models

The features of streamed or transmitted data are extracted from captured data in order to create a model/pattern. This technique is based on mathematic, statistic and probabilistic methods. For this, data distribution in the area of interest is considered to extract knowledge from historical data using techniques such as time series analysis, multivariate analysis and Bayesian statistic. Jin and Nittel [10] propose a technique to detect events by setting up an event boundary. This approach utilises a moving average technique and a statistical model. The moving average technique is performed to reduce noise. The main contributions of this work compute a noise variance value and a threshold.

This probabilistic technique can be written as fractions, decimals or percentages. An output value is between zero and one, [0, 1]. Bayesian networks apply probabilistic technology to create a model which can capture uncertain knowledge. They are represented as a directed acyclic graph (DAG). It has been used for many event detection applications. Fire risk detection is proposed by Daniela et al. [11] using this technology by learning an environment and change a sensing frequency considered by risky areas. Nodes go to sleep mode when they are in a low risk area/situation in order to save power consumption, in contrast, the environment sensing is activated into higher frequency when it is found that the sensing areas are risky. Yin et al. [12] propose a model called Dynamic Conditional Random Field (DCRF) for a Spatio-Temporal Event Detection (STED) using probabilistic technique to discover event in time and space. The probabilistic model is created from learning processes in order to find a relation of a spatial sensor data in each time slice (a sensor data is
considered in a time framed) and a temporal relation between neighbour nodes across time slices. The occurrences of an event can be inferred from historical observed data. Sekkas et al. [13] propose two levels of fire detection. An optimised cumulative sum technique is utilised at first level using temperature and humidity sensor data in each node. When the occurrence of fire is detected in this level, probability outputs are fed to the next level. In this level, Dempster-Shafer (D-S) theory is adapted to fuse the probability output from nodes with vision sensors.

2.3.2.2 Tree Structure

A tree structure is a model to describe a relation between attributes for multi-variable analysis. Learning processes create the model partitioning entropy of attributes from a training dataset. The over-fitting problem is considered when the model is constructed and it can be solved by data dimension reduction. Moreover, the flexibility of model adaptation after installation is low when it requires rebuilding the model after installation and consumes lots of resources to perform this task. The sequence of tree path conditions or rules distinguish phenomenon or classify events. The model is applied to process an event in attributes from tree nodes which can be described as $f: \{X\} \rightarrow y$ where $X$ is the set of attributes leading to class $y$. A matching process on the tree has an issue when there is a big tree or many attributes because it needs more energy to process this task especially when the searching process is time consuming. Bahrepour et al. [14] propose a pruning technique which is the tree size reduction technique. It can manage incomplete sensor data to cope with missing sensor data readings. Ortmann et al. [15] propose relational tree structure against available capability or a threshold with logic based events classification.

2.3.2.3 Fuzzy Logic and Association Rules

Fuzzy rules utilises association rules to define event patterns. Association rules consist of two main components - antecedent and consequent. If/then statements are applied with criteria to identify the relations between variables. This technology is not based on precise thresholds (the value can be between high and low, hot and cold) in order to make a decision in observed data so the
result is robust and leads to a tolerant system. For example, if the rule defines that the speed 120 kilometres per hour (and above) is fast so 119 kilometres per hour would be detected as slow; but fuzzy value (crisps) can determine this value is not in the class of slow speed. However, rules’ combination is growing fast so rule’s size is an issue to be considered when there are many features and attributes in a system. These rules are installed in a memory in a node which is small and it needs more power consumption in matching process to find an event by comparing with every criteria, the computation time takes longer. A rule reduction technique is able to make the rules’ combinations smaller but then the accuracy drops which is not suitable for some systems so a trade-off between the number of rules and computation time is considered with an acceptable accuracy and response time. Kapitanova et al. [16] propose fuzzy rules by reducing the combination of the rules and show incomplete rules technique for event detection. Liang and Wang [17] propose double sliding windows with fuzzy logic to detect an event. Bostan-Korpeoglu et al. [18] propose the Fuzzy Petri Net (FPN) model for sensor data on an active database at the base station so an overall detection in the area of interest can be processed thoroughly (global vision). Kieu-Xuan and Koo [19] utilise fuzzy rules to prioritise events in each cluster in networks and a decision is made at a fusion centre. In conclusion, the flaw of fuzzy rules based techniques is a delay from matching processes since the number of rules is dramatically raised when the features are increased.

2.3.2.4 Pattern Based

Pattern analysis from streaming sensor data has been applied for event detection; pattern defining and pattern matching. For this, sensor data behaviour is observed in order to generate models. Zhang et al. [20] propose an event detection technique from infrequent occurrence patterns within a timeline. Frequent patterns are pre-defined from learning process. A signal is sent to a base station when a pattern is detected. Imran and Khan [21] propose Identifier-based Graph Neuron (IGN) for in-network pattern recognition. Patterns are stored in a database manually. A historical data is contained in associate memory for a comparison between patterns to find a local event and then a consensus technique is performed for a global processing. A decision-
making is not performed at the top of the graph and only matched nodes go to active state, the others are in the idle state which leads to energy saving from the changing state and a communication overhead. Wittenburg et al. [22] adapt pattern recognition for in-node application-specific events. Event characteristics are discovered in pattern recognition processes. Features of data are extracted from training processes and reduce the data dimensions. The learning model and the descriptive/significant data are exchanged. In classification process, nodes exchange only processed data from the former processes. The drawback of this approach can be a delay from waiting neighbours signal which is not considered. Moreover, there is no limitation time of this waiting time. The minimum number of exchanged data or/and the collecting time should be defined to control the response time. Jin et al. [45] propose anomaly detection using Symbolic Dynamic Filtering (SDF) technique. It uses feature extraction for pattern identification from sensor time series to enhance the classification performance. This technique compresses a data and reduces number of classes. Euclidean distance metric is utilised to classify features.

2.3.2.5 Hybrid Methods for Event Detection

An integration across technologies can enhance performances in different dimensions, namely accuracy, computation, energy consumption and response time. Automated methods possibly cooperate with predefined patterns or between automated technologies. Alexandra et al. [23] use a decision tree technique to detect the number of people in the laboratory from sensor data and manual collection dataset. Zoumboulakis and Roussos [24] propose four functions for event detection. First – suffix array multiple-patterns matching – it is performed to convert sensor data into string data and then this data is processed in order to detect an event. Pruned suffix array is utilised to improve the performance in searching process to find an occurrence pattern from binary tree, therefore a computation becomes better as the pattern becomes bigger. Second – known pattern matching – an expert user predefines the model and then compares with sensor reading data using a distance-based algorithm. Third – unknown pattern matching – a model is created from a learning process using a normal situation. An occurrence is detected when the normal pattern and data reading cannot be matched. Pattern matching algorithms can reduce
the power consumption using Dynamic Sampling Frequency Management (DSFM) at the CPU active state. Finally – Markov Chain probabilistic detection – it is utilised for abnormal data or event detection. The probabilities are generated in learning process from the data frequencies and transition. Amin and Khan [25] propose pattern recognition utilising an adapted Distributed Hierarchical Graph (DHGN) algorithm, DHGN dual-layer. The first layer, a sensor node determines recall/store status and then output from this layer is sent to an upper level. A sensor data is compared with neighbours for learning and then a pattern is stored in nodes. An occurrence is detected by comparing with a threshold. If a sensor data meets the threshold criteria and match the stored pattern, the output from this level (node id, time stamp, class id) is sent to the base station for processing in a higher layer. The second layer, voting technique is utilised at a base station with Simulator/Interpreter (SI) module. Xue et al. [26] propose a spatio-temporal contour map matching technique that considers accuracy and power consumption. User patterns can be added to specify event types through SQL and matched with contour maps. Multi-path routing (map construction) is performed in order to reduce the frequency of transmission packets. Linear regression is utilised for merging contour regions and contour compression to reduce the partial maps size. Hashemi and Yang [27] propose a flexible decision tree algorithm (FlexDT) with concept drifts and guards. Fuzzy logic and sigmoidal function is utilised in order to increase noise robustness. Thomason and Parker [28] propose Probabilistic Suffix Tree (PST: a tree structure with conditional probabilities) and symbol compression for distributed hierarchical anomaly detection. The raw sensor data is classified into groups and finds the level of anomalous data over time using Fuzzy Adaptive Resonance Theory (ART) neural network. Semantic data is extracted from the symbol compressor and then an anomaly processing is performed using PST which is adapted from Variable Memory Markov (VMM) model. Cao et al. [50] show two principle methods for outlier detection in large data stream from vary window size including minimal probing and lifespan-aware prioritisation (LEAP). Minimal probing (KNN_MinProbe) is performed to check a minimal needed evidence from neighbours. A similar KNN metadata structure is stored and maintained per sliding window to reuse in the next window and only counted neighbour is considered for outlier candidate. LEAP is utilised to find an impaction of an earlier outlier from earlier window and a new window.
2.3.2.6 Clustering

Clustering is an unsupervised learning technique which is not trained from a labelled dataset. Similar objects are grouped whereas dissimilar objects are distinguished. Similarity is defined from a distance measurement such as Euclidean, Cosine, Jaccard and Edit distance. The number of clusters need to be provided and it consumes lots of power for a cluster processing including finding the appropriate number of clusters, moreover, it requires more memory than classification techniques. Bahrepour et al. [29] show that adapted k-means (modified k-means with Manhattan distance) is able to detect unknown event patterns which improves response time and solves the problem of standard k-means, however, the measurement results, the accuracy and false alarms are not good. Guo et al. [30] propose sensing models from distances metric between nodes; a sequence-based. Orders of distances are sorted and then sensing sequences are compared with the original sequences in order to discover the source of abnormal sensor nodes.

In terms of technology utilisation from aforementioned techniques can be summarised as shown in Table 2.1. These works have different strategies in order to detect an event in WSNs. This table is shown event detection techniques and global processing at a gateway/powerful node. The works from [6], [7] and [8] are low an automated system/user dependence which are not suitable in real-world applications. With the decision tree techniques [14] and [15] have a cost when the models want to be modified because all the tree structure needs to be rebuilt. Fuzzy logic is a tolerant technique from threshold defining; [16], [17] and [19]. The works in [17] and [19] are not considered with the number of rules which can affect the delay from a matching process. The approaches from [20] and [21] perform pattern based without pattern size consideration which leads to additional cost from memory usage and matching process. Moreover, it can be the cause of increasing energy consumption when the processing time is longer. In [22] data dimensions are reduced but this algorithm does not control response time. Hybrid approaches enhance the detection performance, however, simple techniques for processing in wireless sensor node are important since a node has resource constraints. More transmission and processing in this level can reduce the node lifetime. Clustering based event detection in [29] and [30] can detect an unknown event but a resource demand from these techniques is
high which is not suitable for processing in a node. However, density-based clustering is another choice for unsupervised learning, it does not have a pre-defined cluster but it is suitable for a low dimensional data. From these techniques it can be summarised by groups of technologies as in Table 2.2.

Table 2.1: Event processing techniques

<table>
<thead>
<tr>
<th>Processing techniques</th>
<th>Solutions</th>
<th>Use a gateway</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-level threshold [6]</td>
<td>Threshold</td>
<td>Yes</td>
</tr>
<tr>
<td>Composite event detection [7]</td>
<td>Threshold</td>
<td>No</td>
</tr>
<tr>
<td>Shapes based [8]</td>
<td>pattern (manually)</td>
<td>No</td>
</tr>
<tr>
<td>Event boundary [10]</td>
<td>Statistic</td>
<td>No</td>
</tr>
<tr>
<td>Dynamic conditional random field : DCRF [12]</td>
<td>Probability</td>
<td>No</td>
</tr>
<tr>
<td>Two levels fire detection [13]</td>
<td>Probability</td>
<td>Yes</td>
</tr>
<tr>
<td>Pruning tree [14]</td>
<td>decision tree</td>
<td>No</td>
</tr>
<tr>
<td>Relational tree [15]</td>
<td>decision tree</td>
<td>No</td>
</tr>
<tr>
<td>Rule based reduction [16]</td>
<td>fuzzy rules</td>
<td>No</td>
</tr>
<tr>
<td>Double sliding window [17]</td>
<td>fuzzy rules</td>
<td>No</td>
</tr>
<tr>
<td>Cluster event priority [19]</td>
<td>fuzzy rules</td>
<td>Yes</td>
</tr>
<tr>
<td>Infrequent patterns analysis [20]</td>
<td>pattern recognition</td>
<td>Yes</td>
</tr>
<tr>
<td>Identifier-based graph neuron : IGN [21]</td>
<td>pattern recognition</td>
<td>Yes</td>
</tr>
<tr>
<td>Adapted pattern recognition [22]</td>
<td>pattern recognition</td>
<td>No</td>
</tr>
<tr>
<td>Number of people detection [23]</td>
<td>decision tree + user</td>
<td>No</td>
</tr>
<tr>
<td>Four function model [24]</td>
<td>pattern recognition + decision tree + probability</td>
<td>No</td>
</tr>
<tr>
<td>Distributed hierarchical graph : DHGN dual-layer [25]</td>
<td>pattern recognition + threshold + statistic</td>
<td>Yes</td>
</tr>
<tr>
<td>Contour map matching [26]</td>
<td>manual pattern + statistic</td>
<td>No</td>
</tr>
<tr>
<td>Flexible decision tree : FlexDT [27]</td>
<td>decision tree + fuzzy rules + probability</td>
<td>No</td>
</tr>
<tr>
<td>Probabilistic suffix tree : PST [28]</td>
<td>decision tree + fuzzy rules + probability</td>
<td>No</td>
</tr>
<tr>
<td>Unknown event detection [29]</td>
<td>Statistic</td>
<td>No</td>
</tr>
<tr>
<td>Sequence-based [30]</td>
<td>Statistic</td>
<td>No</td>
</tr>
</tbody>
</table>

Some works perform both classification and clustering into the detection system. Yin et al. [44] propose a two phase of event detection using classification and clustering techniques. The classification model (OneSVM) is utilised for filtering normal activities while the second phase utilised the clustering method (KNLR) for abnormal activities analysis.
The processing at a gateway node/base station is able to enhance the performance. However, the communication between nodes and gateway has to be optimised from receiving redundant signals and a trigger/signature is sent to a gateway node, only important attributes are in the packet which is sent to the gateway. Moreover, some techniques do not consider the waiting time delay from neighbouring nodes and the minimum nodes for confirmation at this level.

Table 2.2: The performance comparisons between event detection technologies in WSN

<table>
<thead>
<tr>
<th>Technology</th>
<th>Automatic</th>
<th>Adaptive</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predefined value [6],[7]</td>
<td>NO</td>
<td>NO</td>
<td>High false</td>
</tr>
<tr>
<td>Complex model [10],[11],[12],[13]</td>
<td>YES</td>
<td>NO</td>
<td>-</td>
</tr>
<tr>
<td>Tree structure [14],[15]</td>
<td>YES</td>
<td>NO</td>
<td>High cost on tree construction and matching process</td>
</tr>
<tr>
<td>Fuzzy logic and association rules [16],[17],[19]</td>
<td>YES</td>
<td>NO</td>
<td>Rules’ size and number of attributes consideration and delay from matching process</td>
</tr>
<tr>
<td>Pattern based [20],[21],[22]</td>
<td>YES</td>
<td>NO</td>
<td>Pattern size concern and matching process</td>
</tr>
<tr>
<td>Hybrid [23],[24],[25],[26],[27],[28]</td>
<td>YES</td>
<td>NO</td>
<td>Need to be simplified</td>
</tr>
<tr>
<td>Clustering [29],[30]</td>
<td>YES</td>
<td>YES</td>
<td>High computation</td>
</tr>
</tbody>
</table>

2.4 Collaborative Event Processing

Collaborative processing is performed by groups of correlated sensor nodes which collectively exchange data in order to enhance detection performance. As a result no decision is made by a single node; a decision comes as the result of collective decision-making processes. A signal is not sent to user/actuator directly after an occurrence of an event is detected, a signal from other nodes is processed for an output confirmation. The network infrastructure has an important role in detection performance metrics. Collaborative event processing techniques have been proposed as a taxonomy in Figure 2.2 which can be classified in three categories:
- knowledge fusion and aggregation,
- collaborative/consolidated sensing and
- cluster structuring

2.4.1 Knowledge Fusion and Aggregation

The network topology of this category is shown in Figure 2.3. Data is sent between nodes and the base station.
Each node processes an event and then sends all neighbour nodes a signal to determine their own detection in a coverage area. An output from neighbour nodes which detect the same occurrence and try to send the signal is an issue to be considered. There is a transmission cost and a traffic problem when all the detected nodes send the signal at the same period. Moreover, an appropriate number of confirmations is concerned because of delay and accuracy. The waiting time for the confirmation is a cause of delay. The accuracy can be reduced when the number of confirmation nodes is too small so the minimum number of confirmation nodes needs to define carefully. The signal from the local sensing is sent to the fusion centre at the base station which is higher capability for the global view in a same period of an observation. Banaouas and Mühlthaler [31] propose a probabilistic model, Local Fusion Detection: LFD, which considers accuracy and energy consumption. A decision-making is performed when the summation of sensitivity of neighbour nodes is greater than a detection threshold. The parameters for the sensitivity value are including the distance metric of node, event, and noise. Gaussian noise is utilised in order to define the threshold. Sheng et al. [46] propose collaborative sensing to reduce an energy consumption in mobile sensor device. Virtual Sensor Graph (VSG) is presented to represent a moving trajectory of mobile device which can predict the next period of sensing to save energy. Sousa et al. [48] show an interval comparison method using reference values of fault and occurrence of an event from neighbour nodes which is modelled as numerical intervals. The control chart method (CCM) is used to summarise subset measurements from neighbours and confidence interval method (CIM) is performed for event confident from the detection data from neighbouring nodes. This approach is performed only Gaussian distribution data. Wittenburg et al. [49] propose AVS-Extreme platform using machine learning adaptation and perform cooperative fusion classifier for feature classification from distributed nodes by exchange extracted feature and merge them for prototypes creation. Data from malfunctioned sensors are discarded. Dempster-Shafer is utilised in order to increase the confidence levels.
2.4.2 Distributed/Consolidated Sensing

A distributed model is performed to detect an event in an area of interest; signals are propagated in order to complete the final decision-making. A destination for decision-making can be either a sensor node or the base station which is shown in Figure 2.4.

Abadi et al. [32] propose a distributed processing method that utilises external database in order to filter data by conditions and thresholds. The threshold table is split into smaller size tables and stored in nodes to reduce the memory requirement. The related data from external tables and in-node is examined and fused/aggregated. Sensor reading data is compared with the thresholds for each condition (logic) in order to find an event. Moradi et al. [33] propose Bayesian based approach for two hypotheses; normal and event. In-network processing utilises sensor data from its neighbours in order to obtain an estimated data using Kalman estimators to reduce false and unavailable data. Gaussian state-space model is performed in this step. A signal from each related node is sent to upper level in order to make a final decision from global data fusion algorithm (optimum fusion) at a base station. Janakiram et al. [34] propose the technique called event hop-count and it is counted by sensor type. It utilises tree structure and event occurrences from the node. This technique is not suitable for big networks because of the delay of responding from child nodes. L. Peng et al. [35] propose threshold defining technique using Bit-string Match Voting (BMV) that detects abnormal sensor data at bit level gradually and considers bit string trends in nodes buffer when sensor data reaches a threshold. The sensor data
is encoded to bit-string instead of using data patterns before sending to its neighbours in order to reduce the communication cost and then the voting process is performed at the neighbouring nodes. The problem of this technique is how to define the number of neighbours for the voting process. A low number of considered neighbour nodes means less reliability, the delay in response time increases when number of confirmation nodes is too high. D.-H. Tran [36] proposes DFT using signal segmentation and sliding windows to reduce memory usage. Gossip protocol is applied to enhance the performance of redundant messages. These two techniques perform collaborative sensing to obtain accurate results. A. Muhamad Amin and A. Khan [37] propose pattern recognition in graph (Distributed Hierarchical Graph Neuron : DHGN) and collaborative comparison with neighbours. Sub-patterns are created from the main pattern for the comparison process. The collaborative comparison learning is processed from the comparison between adjacent nodes. If the data matches the pattern, the signature data is sent to a base station. Ngai and Xiong [47] propose a technique to observe an environment using mobile phones and sensor nodes by considering a sensing priority which is more in mobile phones. An occurrence of event detected from mobile phones is sent to nodes in that area to perform collaborative sensing. A time interval sensing of an occurrence from mobile phones is checked in order to accumulate a sensing quality in that area.

2.4.3 Cluster Structuring

This network topology is adapted from the two previous structures described above. Since the communication range is one of a sensor node limitations, the communication distance is an issue to be considered. Moreover, as further as a node tries to send a data, power consumption is increased. When a node sends to the gateway a signal, the possibility of every active node in the occurrence area tries to send the same signal, which leads to traffic problems and the average nodes life-time in the network is reduced from unnecessary transmission. Network clustering can reduce inefficiency tasks above by assigning a cluster header which is a representative to communicate to other clusters and/or gateway. This topology is shown in Figure 2.5.
Chapter 2  Background

The network is structured into clusters; one dedicated node (elected following various strategies) called cluster head is allowed to communicate with other clusters - via their respective cluster heads. A group of clusters can be represented as knowledge fusion and aggregation and/or distributed/consolidated sensing. The signal from the sensor head is sent to the base station to process for the next step. Li et al. [38] propose Data Service Middleware (DSWare) for real-time event detection with confidence function. The event is processed as compound events at the cluster head by comparing confidence value with a user-defined threshold. Only the cluster head can send user signals. The confidence value is computed from sub-events in the network which can be a detection result from different sensor types. Each sensor has to predefine a weight depending on application because the dominance of sensors is not the same in different applications. Lai et al. [39] propose publish/subscribe middleware (PSWare) in composite events utilising event temporal information. This approach has three processes namely define events, determine event relationships, and filter event for instant location. Distributed processing and time synchronization are considered. If there is an event occurrence, the trigger is to be sent to neighbours and a neighbour group in order to wake the nodes up for detecting the event with time synchronization. If the nodes detect the same events, the trigger from only one group header is sent to sink.

Figure 2.5: Cluster Structuring
2.5 Discussion

In this chapter we reviewed sensor data processing, wireless sensor networks, event processing and their challenged issues. We also discussed event detection models, and collaborative processing in WSNs. Sensor data is analysed and used to build models for classifying or adding data into logical which characterise an occurrence of events under resource constraints. Decision-making in only one node results in a low accuracy, so a collaborative sensing is necessary for increasing accuracy and reducing fault alarms. Collaborative sensing can be done at the level of the network, in-network and/or at the base station. Machine learning techniques have been studied for event processing in WSNs in order to improve the different aspects like accuracy, response time, traffic, computation and overall performance of the event detection.

The next chapter describes data processing for event detection by utilising machine learning and data mining technologies. They are performed to find an outlier and then detect an event. The outlier processing is done in-node which is then analysed to process an event.
Chapter 3. Data Processing for Event Detection

A sensor node observes environment and sends signals to other nodes when an occurrence is detected. Occurrence detection mostly depends on a value which is over a certain threshold and referred to as an outlier. Events are observed from the combination of sensor data which is based on threshold values. A threshold can be set up manually by expert users or defined using statistical methods based on history data. A changing environment leads to variations in the threshold over time and location. Sliding windows are performed for a time series dataset. A threshold can have only one value, which distinguishes between normal situations and outliers for a primary detection.

In this chapter we describe data preparation for event processing in order to create a model which can lead to a better detection performance. The data is classified and labelled into three groups by an unsupervised learning method. It categorises sensor data into groups automatically. Each group is labelled from statistic data from pre-defined rules. The output from this stage is the labelled data for model creation.

3.1 Experiment Setup

The experiment for this work is applied to sensor data collected from the Smart Campus test-bed at the University of Surrey. The sensor data is then processed to detect an event such as a meeting in a room. Sensor nodes have 10 seconds sampling rate and historical data is collected in a server. Each node senses passive infrared (PIR), light, temperature, audio level and power consumption on the wall. Node identification and sensing time is also included in the record. The sensor data is an integer data, some sensors have to be converted from sensor reading data to a real value such as light and temperature. Computing the temperature in Celsius from sensor data can be converted by $5.3 \times 0.01$.
data reading – 40.4 and computing the light in Watts can be converted by \( \frac{\text{data reading} - 1.9327}{2.4774 \times 10^7} \).

The room for this experiment is a meeting room. It contains 4 sensors in different location on the walls such as beside a window, in front of the room beside a board, opposite windows and back of the room (Figure 3.1). Python is the implementation tool which is chosen for this experiment since it is free, easy to understand and light.

![Figure 3.1: The meeting room layout for the experiment](image)

### 3.2 Sensor Data Pre-Processing

The first step of data analysis in event processing is to classify between a normal situation and an occurrence of an event. An outlier can possibly be noisy, erroneous or event data. Outlier analysis is a vital step in order to filter significant data which leads to the detection performance from its model.

An outlier is defined as an observation which has deviated from other members in the dataset. Johnson also defines an outlier as an inconsistency of an observation data when compared with the rest of the dataset [40].
Many applications apply this detection technology such as military surveillance in enemy activities, credit card fraud detections, cyber security for network intrusions and safety critical systems to detect faults in order to interpret actionable information [41]. The data source for an outlier analysis in event detection in WSNs is collected from sensor nodes as streaming data.

This work utilises and analyses historical sensor data for creating a model to detect an outlier. A classification technique is utilised for model creation so a training dataset is the first factor for the detection performance. This part describes the processes to obtain the training set starting from, the first, data preparation which generates overlapping sliding windows and processes discrete wavelet transform. Second, data processing, to classify data into groups and finally data labelling using statistical rules. These processes are shown in Figure 3.2. The first step is data preparation which processes sensor data into overlapping sliding windows in order to reduce a false detection from a large volume data processing. Sensor node can produce noise so discrete wavelet transform is chosen to process for each window since it can reduce noise. Moreover, a window size is also reduced and it is also a benefit for data processing in a wireless sensor node. K-means clustering processes these windows and categorises data into groups which are separated by their distance. Each group is labelled using statistical values in the next step by pre-calculating these values for the rules. The statistical rules is created from the history data of each sensor so the rules give more realistic than manually threshold defining. The labelled data is utilised for a training dataset to create a model. These steps are shown in Figure 3.3.

Each node senses a phenomenon and measures data individually such as light, temperature, noise level and passive infrared (PIR). The processing of outliers is performed in a node independently. This experiment captures sensor reading

![Figure 3.2: The training dataset processing structure](image-url)
data such as microphone and light in a meeting room with 10 seconds sampling rate. The data is formed as sliding windows with an overlap to an adjacent window in order to reduce faults in a case of an outlier is in between windows. For example, if there is a set of data as

\[ T(x) = 13, 23, 24, 26, 300, 280, 210, 350, 22, 33, 24, 26, 32, 28, 32, 37 \]

and the window size is 4, the data is divided into windows without an overlapping as

\[ T(w=4) = <13, 23, 24, 26>, <300, 280, 210, 350>, <22, 33, 24, 26>, <32, 28, 32, 37> \]

From these non-overlapped windows, a phenomenon cannot be detected because each window can be defined as a normal situation. The overlapping windows is utilised to reduce the fault from this problem. This example gives the overlapping of each window as 2. These windows can be constructed as overlapping windows as below

\[ T'(w=4) = <13, 23, 24, 26>, <24, 26, 300, 280>, <300, 280, 210, 350>, <210, 350, 22, 33>, <22, 33, 24, 26>, <24, 26, 32, 28>, <32, 28, 32, 37> \]

The occurrence can be found in the second window when overlapping sliding windows are applied since a significant different value within the window is found.

Discrete wavelet transform (DWT) is chosen to perform for the primary processing, data preparation, in order to reduce noise and error in each window. Moreover, data size is reduced after processing this function because it also compresses the data to smaller data size which is needed for data processing and resource limited system. The outputs are differentiated into two parts for different filters, high pass filter \((h[n])\) and low pass filter \((g[n])\). High pass filter returns the detail coefficients \((cD)\) and another filter returns approximation coefficients \((cA)\) and they are correlated in each other (Figure 3.4).

This work evaluates DWT functions since they have a different performance to detect an outlier from two datasets (normal and normal with outlier) as shown in table 3.1. In order to select one of them, two streaming window data \((cA)\) are compared using cosine distance technique to find the similarity between them. The output is the list of cosine distance between two data sources from each
window at the same timestamp. The output which is close to zero has more
dissimilarity and the most similarity has a value close to one. A threshold is
utilised to detect an outlier from these streams which is defined as

\[ \text{Threshold} = \text{minimum value of cosine distance} + \text{standard deviation of this distance} \]

Figure 3.3: The process to create training datasets and clustering performance evaluation

Figure 3.4: Discrete wavelet transform's block diagram [42]
This threshold is based on statistical values, minimum value and standard deviation, so it is not fixed from user-defined threshold. This experiment measures the earliest detection and the most distinctive/dissimilarity between normal and outlier in data streams so the position of the detection is considered. The result in Table 3.1 shows that the cosine distance is varied by the DWT functions from the same input data stream. The earliest detected outlier are db1, coif1, rbio1.1 and dmev at the third position. The further consideration is how much dissimilarity between two data streams considering from the cosine distance value. It is found that coif1 can distinguish the data as the lowest cosine distance (the most dissimilarity) when compared with the other function that give the earliest detection so the coif1 is chosen for the further experiment. Coif1 can reduce the data size 33.33% for each window when the window size is 4. However, the data reduction is superior when the window size is higher, 33.33% reduction when the window size is 12 and 48% when the window size is 100.

Table 3.1: Outlier detection comparison between different DWT functions based on the experiment that is completed in this research

<table>
<thead>
<tr>
<th>Function</th>
<th>Position found</th>
<th>Cosine distance</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haar</td>
<td>4</td>
<td>0.14922</td>
<td>0.7642</td>
</tr>
<tr>
<td>Db1</td>
<td>3</td>
<td>0.77329</td>
<td>0.7877</td>
</tr>
<tr>
<td>Sym2</td>
<td>4</td>
<td>0.35976</td>
<td>0.7690</td>
</tr>
<tr>
<td>Coif1</td>
<td>3</td>
<td>0.68150</td>
<td>0.7747</td>
</tr>
<tr>
<td>Bior1.1</td>
<td>4</td>
<td>0.14920</td>
<td>0.7411</td>
</tr>
<tr>
<td>Rbio1.1</td>
<td>3</td>
<td>0.77330</td>
<td>0.8101</td>
</tr>
<tr>
<td>Dmey</td>
<td>3</td>
<td>0.6841</td>
<td>0.7922</td>
</tr>
</tbody>
</table>

The detection performance can be affected by every step of the process such as window size, training dataset and detection model. Various window sizes are considered to analyse the performance of the clusters, completeness score and homogeneity score. This experiment labels the data manually using rules created from statistic values (Figure 3.5). The “range” is defined as “(max-min)/10”. These rules are started from mean. An upper bound of “Avg” rule is computed from mean added by standard deviation/range and lower bound is computed from mean minus by standard deviation/range. The lower bound of “Low” and the upper bound of “High” rule are computed from standard deviation/range multiply by two. The lower bound of “vLow” and the upper
bound of “vHigh” rule are computed from standard deviation/range multiply by three. The length of this level is the triple size from mean which means that the first time of standard deviation/range is for average rule (Avg), the second time of standard deviation/range is for low/high rule (Low/High) and the third time of standard deviation/range is for very low/very high rule (vLow/vHigh). These rules have conditions for the lowest value and the highest value in order to cover all of raw data in the dataset since the three time of standard deviation from mean, lower bound of vLow and upper bound of vHigh, is not guaranteed to cover all the member in the dataset. These rules are performed for a further process that is manual data labelling for cluster analysis.

```
1   vLow = [mean-3*std if mean-3*std < min else min, mean-2*std]
2   Low = [mean-2*std, mean-std]
3   Avg = [mean-std, mean+std]
4   High = [mean+std, mean+2*std]
5   vHigh = [mean+2*std, mean+3*std if mean+3*std > max else max]
```

Figure 3.5a: Rules for data labelling using statistic values varied by standard variation

```
1   vLow = [mean-3*range if mean-3*range < min else min, mean-2*range]
2   Low = [mean-2*range, mean-range]
3   Avg = [mean-range, mean+range]
4   High = [mean+range, mean+2*range]
5   vHigh = [mean+2*range, mean+3*range if mean+3*range > max else max]
```

Figure 3.5b: Rules for data labelling using statistic values varied by data range

Figure 3.5: Rules for data labelling

Data is categorised into three groups using k-means clustering technique since the processing time and the resource requirement of k-means processing is acceptable. These groups from k-means are assigned as low, medium and high. The clusters are analysed using manual data labelling from the rules which is described before. The two sets of statistical rules (standard deviation and range) are reformed into three groups in order to perform for clustered data. The first group is assigned from “vLow” and “Low” rule for the low group, the second group, medium, has only “Avg” rule and the last group, high, is assigned from “High” and “vHigh” rule. The clustered performance is evaluated and compared in different overlapping sliding window size as shown in Table 3.2 which is plotted as a graph in Figure 3.6. The overlapping sliding window is represented
as \((x,y)\) while \(x\) is a window size and \(y\) is a last point which is not in a member of a next window which means that the next window is started from \(y+1\) of the current window. This experiment is focused on how window size affects to the cluster performance but the size of an overlapping is not included in the result. It is determined by observing the data manually. The overlap size at 10 is chosen by considering from the raw sensor data and it is enough for the experiment in this dataset. The larger overlap size would give the better result since the data is processed thoroughly from different windows however the processing load is more than the smaller overlap size because the number of window is increased. Completeness score and homogeneous score are the performance measurement of the clustering model. The completeness score is between 0 and 1 and all the data points which are assigned in the same class are in the same cluster. It means that the perfect cluster is created when all data points are in a single cluster and the score is 1. Homogeneous score is also between 0 and 1 and all data points are not members for the same class. It means that the perfect cluster is created when all data points are separated and the score is 1 [43]. The result shows that the rules using range value has a better performance than using standard deviation and the best window size is \((100,90)\). The first value of the window \((100,90)\) is the size of each window and the other is the started data point for the next window.

<table>
<thead>
<tr>
<th>Window size</th>
<th>Completeness score</th>
<th>Homogeneous score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Std</td>
<td>Range</td>
</tr>
<tr>
<td>(25,15)</td>
<td>0.419</td>
<td>0.786</td>
</tr>
<tr>
<td>(50,40)</td>
<td>0.418</td>
<td>0.791</td>
</tr>
<tr>
<td>(75,65)</td>
<td>0.408</td>
<td>0.799</td>
</tr>
<tr>
<td>(100,90)</td>
<td>0.407</td>
<td>0.804</td>
</tr>
<tr>
<td>(125,115)</td>
<td>0.419</td>
<td>0.790</td>
</tr>
<tr>
<td>(150,140)</td>
<td>0.406</td>
<td>0.764</td>
</tr>
</tbody>
</table>

The clustered data from k-means clustering and manual labelled data from the rules for three groups is compared as graphs in Figure 3.7. It represents two members in a window, x-axis represents the first member value and y-axis represents the third member value from each window. The graph shows an
overall image of how statistical rules and cluster data can group and categorise data into 3 groups. In average, they are in the same direction, only some data points are different. It can be related to the results of the cluster analysis of manually rules labelling and grouping by k-means clustering which means that there is some faults of data clustering and/or rules performance so the cluster evaluation is not a perfect clustering, it is not equal to 1. The output from clustered data is an input for creating a classifier model. The processes which are described above are the technologies selection and the data preparation methods. The next section describes the details in outlier detection.

![Completeness score](image1.png)

Figure 3.6a: Completeness score

![Homogeneous score](image2.png)

Figure 3.6b: Homogeneous score

Figure 3.6: The completeness score (a) and homogeneous score (b) of the cluster in varied window size of streaming sensor data
Chapter 3  Data Processing for Event Detection

3.3 Cluster Data Processing

The process of this part of the experiment is shown in Figure 3.8. The first step is data preparation which has two main processes including sliding windows (overlapping) and discrete wavelet transform. Sensor data is constructed as overlapping sliding windows with the size of 100 as

$$W = [w_1, w_2, \ldots, w_{100}]$$

The overlapping sliding windows ($W$) is used for clustering and labelling. The k-means clustering classifies the window data into three groups. Before the clustering, each window is pre-processed using discrete wavelet transform which can reduce noise and data size. The output from this step includes low pass filter ($W_{cA}$) and high pass filter ($W_{cD}$). The window size is now 52 which is the result after the discrete wavelet transform that is 48% data size reduction for each window.
The next step is data clustering from $WcA$ (window data from low pass filter: $cA$). The output from clustering are the indexes of their groups for each window.

This example shows that window 1 and 2 ($WcA_1$ and $WcA_2$) are in the same group (group number 1) and window n is in another group (group number 2). They are grouped based on the similarity of the data but there are no labels to represent the meaning for each number. The average value of each cluster is computed to find three levels of data group namely “low”, “medium” and “high” ($L$, $M$, $H$) in order to label the k-means cluster.

The last step is data labelling which is defined by the rules from raw sensor data aforementioned before. The average value of each window is computed in order to identify the label in each window by comparing this value and the rules such as $W [H, H, L, ..., M]$ for $W [w_1, w_2, ..., w_n]$ and $idxK$-means is $[1, 1, 0, ..., 2]$. The labelled data is transformed into indexing form by comparing between labelled cluster ($L$, $H$ and $M$ which represented in cluster 0, 1 and 2 consecutively) and labelled window $W [H, H, L, ..., M]$, for this it is indexed as $idxOrg [1, 1, 0, ..., 2]$. The indexing of k-means cluster and manual data labelling from rules is utilised for cluster performance evaluation, completeness score and homogeneous score. The process in Figure 3.8 can be separated into two sub-processes, data labelling and data clustering as a procedure below.

A. Reading a raw sensor data

B. Making an overlapping sliding window with the size (100, 90)

1. Data labelling
   1.1 Create statistical rules from raw sensor data (A.) in order to define the rules into 5 different groups (very low, low, medium, high and very high)
   1.2 Compute an average value of each window (B)
   1.3 Classify each window which is represented by the average value (1.2) with the rules (1.1)
Chapter 3  Data Processing for Event Detection

- The rules are classified data into 3 groups (Low, Medium, High)
- The data is represented as idx ['H', 'H', 'L', ..., 'M']

2. Data clustering

2.1 The overlapping sliding window (B) is processed discrete wavelet transform for each window
- The output including cA and cD which are low pass filter and high pass filter

2.2 Cluster data (cA) into 3 groups using k-means algorithm

2.3 Compute an average value of each group

2.4 Define each cluster into a meaningful cluster as low, medium and high by comparing an average values in 2.3
- each cluster window has label as ‘L’, ‘M’ or ‘H’

C. Indexing window data (1.3) relates to the cluster data (2.4, the cluster from 2.4 can be unordered as ‘L’, ‘H’, ‘M’) as ‘0’, ‘1’ or ‘2’

D. Analyse k-means clustering from 2.2 and data labelling manually using statistical rules from C

Figure 3.8: The processes of training data preparation

The result of this part of the work produces the training set with labelled for a classifier creation, classification model.
3.4 Summary and Discussion

The work described in chapter 3 relies on sensor data and runs an unsupervised machine learning (k-means clustering) to assign the window pattern to groups (i.e. clusters). We then use a set of statistical and rule-based analysis to assign labels to each cluster. This part of the work is mainly an offline process. However, in the WSN we require online event detection and analysis.

We use the results of the clustering and labelling to train a classification method that can work as an online system for detecting an event. In other words, the first part of the system uses an automated mechanism to generate the training dataset for the classification method. The training set, obviously, is enhanced by statistical and rule-based analysis, which is a semi-automated method, to assign labels to each cluster and to improve the results of the labelling.

The final result from this chapter is the training dataset which is created from the sub-processes. Each step is evaluated in order to enhance the performance, starting from sensor data selection which is used to create rules for labelling data. The window size is also another factor to be considered. Discrete wavelet transform can reduce noise and data size but the functions give us different results from their processes which we can evaluate in a different perspective for example size reduction, an earlier detection and the complexity of the function. The clustering groups the similarity of data which can be distance-based or density-based clustering. Density-based clustering does not need to define the number of clusters but in this experiment we want the specific cluster number.

It is very time consuming to adjust parameters to get the right cluster number if the density-based is used. Moreover, the computational complexity is very high which is not suitable for sets of data. K-means is a distance-based clustering that we can define the number of clusters and the computation complexity is not too high so the run-time is not too high. However, the work in this chapter has been completed as an offline process so the historical data is not up-to-date and affects the rules which are used for labelling the data. Some parameters of the rules should be changed over the time period to enhance the detection performance.
Chapter 4. Event Processing

An event detection normally processes more than a node for collecting and processing observed data before a decision is made to classify an occurrence as a regular event or as a noise/error. Single node processing can be faulty or biased by environmental changes and therefore more data from different distributed sources may be required. Collaborative sensing appears to be a more robust solution as it processes sensor data from distributed nodes which have an occurrence at the same period of time to confirm/disconfirm the decision making. Information from neighbouring nodes is considered instead of making a decision from a single node. Many approaches make an event confirmation by consulting neighbours before the decision-making in order to reduce the amount of false alarms. Motivated by this, our aim is to research (collaborative) event detection by combining rule based statistical methods and machine learning algorithms, which learn the environment situations from historical data to create a model.

This chapter describes supervised learning in order to create a model/classifier of an environment from sensor data. The labelled data or training set is the result from the previous chapter. The classifier which is created from classification technique is used for observing an environment and performs further processing when an outlier is detected in order to process the event in that data.

4.1 Outlier Processing

The event data can be defined by thresholds, rules and/or models which can distinguish normal situation and occurrences. Machine learning creates models automatically from its historical data which leads to detection enhancement. Machine learning techniques, classification and clustering, classify data features from a streamed/transmitted data. The model creation runs as an offline process and is utilised to observe an environment in a real-time system.
The observed data and the model is processed to detect an outlier within that stream. When the outlier is detected, further analysis is performed in order to find an event in that data stream (Figure 4.1).

![Figure 4.1: Event detection processes](image)

This experiment has an overall process presented in Figure 4.2 which includes online and offline processes. A model is generated from learning stage in offline processes and then this model is utilised to detect an event in online processes. There are two steps in this part, data preparation and environment understanding. The previous chapter describes data preparation using a semi-supervised learning method. The output from this step is utilised to create a model from the environment understanding process by learning the data from the previous chapter. A semantic of sensor data is processes in the online processes from streamed data by a pattern matching process, comparison between data patterns and a model. When an outlier is found, further process, outlier analysis, is performed in order to categorise between noise/error and an occurrence which can be an event that is then sent by a signal to the user/actuator. After the outlier is detected, a collaboration processing can be performed in order to enhance a detection performance instead of single node processing which can produce a false. The sensors share that detection and process together with its own detection that can give a reliability of the result. The collaborative sensing can be performed from conditional statements of sensors depend on an application. A detection of a meeting in a meeting room
is one of the application can be applied for this sensor data. A performance can be evaluated from defining different recall of the detection. An evaluation and result for the collaborative sensing will be performed for a future work.

![Figure 4.2: Event processing model](image)

### 4.1.1 Classification Analysis

Historical sensor data is a crucial element to determine a data model for a learning method in order to understand an environment behaviour. The classifier model, assigns a dataset into classes that consists of training and classification. This experiment pre-processes a training dataset in the previous chapter, data labelling using semi-supervised learning. The dataset classes become known and then a classifier is created on the basis of this data. The classification process utilises the classifier and then processes a new unlabelled dataset to determine its membership for each class.

Classification method is performed in order to identify environment characteristics/classes (e.g., temperature levels – very low, low, medium, high and very high). The training set is fed into this function to create the classifier and then another streaming data, test set, is created in order to evaluate the classifier. In our experiment we split data 60% and 40% for training and testing stage. Since the data is converted from a discrete wavelet transform function then the data is changed. Inverse discrete wavelet transform is utilised to
convert such data into reading data before creating the models/classifiers from this sensor data in order to compare with sensor reading data (Figure 4.3) for online occurrence detection which will be described in the next step. The accuracy results from different classifiers/models are evaluated which are shown in Table 4.1. The three different sensor datasets (temperature, light and microphone) are evaluated. The results shown that the accuracy from every sensor is in the same direction for a same classifier. However, the classification accuracy from this experiment is very high which could be due to an over-fitting problem. This problem comes from the divided dataset for training and testing set so these datasets are very similar and lead to a tight model which reflects the weirdness of a nearly perfect result and cannot be the right evaluation for the real environment. This problem is unavoidable if the training and testing dataset is created from the same set even it is randomly separated.

![Figure 4.3: Classification processes](image)

Table 4.1: Classification accuracy for different models from three datasets (temperature, light and microphone) using divided data of a training and testing set causing an over-fitting problem

<table>
<thead>
<tr>
<th>Classification</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Temperature</td>
</tr>
<tr>
<td>Gaussian Naïve Bayes</td>
<td>97.66</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>94.79</td>
</tr>
<tr>
<td>Random Forest</td>
<td>98.96</td>
</tr>
<tr>
<td>Extra Tree</td>
<td>98.70</td>
</tr>
<tr>
<td>Nearest Centroid</td>
<td>99.80</td>
</tr>
<tr>
<td>Nearest Neighbour</td>
<td>98.18</td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>98.96</td>
</tr>
<tr>
<td>Bagging Meta-estimator</td>
<td>98.96</td>
</tr>
<tr>
<td>Ada Boost</td>
<td>92.97</td>
</tr>
<tr>
<td>Gradient Boost</td>
<td>97.04</td>
</tr>
</tbody>
</table>
A new test dataset is performed from a different dataset of a training dataset which is created from a new streamed data. A training dataset is the result from cluster and labelling data. The new testing set is created manually by setting up the boundary from the rules described in the previous chapter. These rules give an acceptable result with low variance data – an average value and median value are not much different such as temperature data from the experiment, while light and microphone data cannot use the same rules of low variance data because the statistical values (average, range and standard deviation) are misleading affecting the rule’s boundaries. They are redefined by new rules which are computed from pre-processed data in order to adjust some statistical values by removing an over-peeked of data. However the maximum value for the rules is still referred from the raw sensor data in order to set the rules covering the whole dataset. The over-peeked data removal procedure can be described as the procedure below:

1. Sort data
2. Compute standard deviation (std)
3. Set the current position (cur) at rounded up std
4. Compute average value (avg) from cur-1 to [(cur-1) – rounded std]
5. Subtract data between cur and avg (diff)
6. Compare between 2. and 5. (std and diff)
   4.1 if diff < std and cur is not the last data
      Do a right shift of cur
      Go to 4
   4.2 if diff >= std and cur is not the last data
      Set the highest value at cur-1
      Stop
   4.3 if cur = the last data
      Set the highest value at cur
      stop

For example, the sorted data is

12, 12, 12, 14, 14, 20, 22 and the standard deviation is 4.14 so the current position is 5.
The first step is to calculate the average data at point one until four (integer of standard deviation from dataset) to compare with data point five as

12 , 12 , 12 , 14

Average = 12.5    compared data/current position

the differentiate between average value and compared data is 14 – 12.5 = 1.5 which is less than standard deviation so the next data point is chosen to compute the same method and the data to compute the average value is a step of right shift as

12 , 12 , 12 , 14 , 14 , 20 , 22

the difference is 20 – 13 = 7

Average = 13    compared data/current position

The difference between average and compared is now more than standard deviation (7 > 4.14) so it is halt at this point and then the statistical values are recomputed. In this case the data which is selected to compute the statistical values for the rules is only data point 1 to 5, point 6 and onwards are removed since it is suspected to be noise or over-peaked value which misleads the statistical values for the rules so the new standard deviation is 1.09 (computed from the position 1 to 5). This process computes new statistical values however the maximum value (22) is still kept for the upper bound of “very high” of the rule in order to cover all member in the dataset. The two stages of pre-processed data (sorting and over-peaked data removal) are performed for rules creation from the historical sensor data. Figure 4.4 shown the changes from raw sensor data to sorted data and from sorted data to over-peaked data removal).

The raw sensor data also has abnormal data which can be easily noticed in the light and microphone data. These datasets are sorted and it can then be noticed how different they are. The temperature data gradually changes whilst the others are different when they go to a higher value. The last step is used for analysing the dramatically changing data point from sorted data. From the experiment, the range of raw data and processed data for temperature data is nearly the same (about 1150 – 1300) because the variance is not too high. In contrast, the raw data from light and microphone has the different range so the
dramatic changed data point is computed and the new range for computing statistic data is updated. The range of light is changed from about 0 – 2700 while the new range is from about 0 – 1050 and it can be noticed from the light sorted data that the value around 1000 goes higher very fast. The microphone data has the same characteristic of light sensor. The range for computing statistic value is changed from 0 – 140 to about 0 – 21.

![Graphs showing data range adjustment](image)

Figure 4.4: The output graph from each step in order to adjust data range for statistical values which are used for rules creation

The new rules are shown in Figure 4.5 which are considered from average value for the average boundary (avgRange). The first line is a minimum value from the over-peaked data removal but the maximum value takes the value from the maximum of raw sensor data in order to cover all data points (line 2). The average value (line 3) is computed from over-peaked data removal and ceiling the average value because a sensor produces integer values. Since some datasets have a very low mean, the ceiling function can help to make a bit more
length between minimum data and mean for rules creation. The range is considered into two parts because some of the dataset is not in a normal curve which has the mean far away from the median. The first part is a lower range to mean (rangeLower, line 4) which is the differentiation between mean and minimum value divided by three. This means that the range is considered from minimum to mean and is separated into three rule groups, “very low”, “low” and “medium”. On the other side, the range is computed from mean to maximum value (rangeUpper, line 5) which is also separated into three groups, “medium”, “high” and “very high”. This method creating two different ranges (rangeLower and rangeUpper) can reduce an “out of data boundaries problem” which can possibly happen when the average value is very close to the minimum (light and microphone) or maximum value. The dataset which has the mean close to median (temperature) is not affected by the out of scope problem and it can apply the same range for every rule. An example of out of scope boundaries is when the range data is 10, average is 12, maximum data is 100 and minimum data is 0. The rules for “low” and “very low” never appear because the rules’ boundaries are out of scope and below zero which cannot be found in the dataset (the minimum is 0) and leads to a low accuracy of result.

```python
1 minData = min(data)
2 maxData = max(raw_data)
3 meanData = ceil(mean(data))
4 rangeLower = (meanData-minData)/3
5 rangeUpper = (maxData-meanData)/3
6
7 vlowRange = [minData, int(meanData-rangeLower)-2]
8 lowRange = [int(meanData-rangeLower)-1, int(meanData-0.5*rangeLower)-1]
9 avgRange = [int(meanData-0.5*rangeLower), int(meanData+0.5*rangeUpper)]
10 highRange = [int(meanData+0.5*rangeUpper)+1, int(meanData+rangeUpper)+1]
11 vhighRange = [int(meanData+rangeUpper)+2, maxData]
```

Figure 4.5: Rules for data labelling using adjusted dataset

The rules are created from statistical data in which it is possible to have a non-integer value whilst sensor data is integer so every statement that can be non-integer is converted to integer. The first consideration is line 9 (avgRange) which is for the group of medium dataset, “medium” rule. It is started from mean and computes a boundary using rangeLower and rangeUpper for lower boundary and upper boundary by expanding the range from mean for a half value of rangLower and rangeUpper. A group of the low dataset is categorised from line
8, lowRange. An upper boundary of lowRange is calculated from the lower boundary of avgRange - 1 in order to avoid the same value for different rules. A lower boundary is also calculated from rangeLower but it is not divided by two because the first half of the rangeLower is for avgRange. A vlowRange from line 7 is for very low data. The upper boundary of vlowRange is computed from the lower boundary of lowRange – 1. The lower boundary of this rule is the minimum value from over-peaked data removal dataset. On the other hand, the rules that are created for the groups of data higher than avgRange is performed by the same concept as the data groups lower than avgRange. The highRange in line 10 for the group of high value data is similar to lowRange and the vhighRange in line 11 for the group of very high value is similar to vlowRange except the upper boundary of vhighRange is from the maximum value from the raw dataset.

In conclusion of the new rules creation from over-peaked data removal, these rules are created from statistical values (mean and ranges) by considering the out of boundaries problem. The rules are defined by starting from mean. Mean and median can be different (small or large different depends on the nature of sensor data) so the range values are performed into 2 parts, under and upper mean, in order to solve this problem such as a minus value that cannot possibly be a member of sensor data.

These rules are created for data labelling for a training and test set. They are performed to label the clustered data which is described before for the training dataset while the test set is labelled and grouped manually by these rules. The data labelling for training and test set is processes separately (they are from different sources of the data – sensor reading data and random data respectively) because of the over-fitting problem. The statistical data is computed for each sensor and creates the boundaries for three classes including “low”, “medium” and “high”. The test dataset utilises these rules to create and label data followed by the rules’ boundaries for every class by defining the value in class “low” from vlowRange and lowRange. The data of class “medium” is created from avgRange and the data of class “high” is created from highRange and vhighRange. The data for the test dataset is 2200 which has 25 windows for each sensor and the data is created randomly within the rule boundaries. The classifier performance in terms of accuracy from the new rules and manual
labelled test dataset is shown in Table 4.2 and Figure 4.6 and the performance in terms of computational time is shown in Table 4.3 and Figure 4.7.

Table 4.2: Classification accuracy for different models from three datasets using over-peaked data removal and separated training and test set by labelling the test set manually from the rules’ boundary

<table>
<thead>
<tr>
<th>Classification</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Temperature</td>
</tr>
<tr>
<td>Gaussian Naïve Bayes</td>
<td>81.82</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>77.27</td>
</tr>
<tr>
<td>Random Forest</td>
<td>77.27</td>
</tr>
<tr>
<td>Extra Tree</td>
<td>81.82</td>
</tr>
<tr>
<td>Nearest Centroid</td>
<td>81.82</td>
</tr>
<tr>
<td>Nearest Neighbour</td>
<td>81.82</td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>81.82</td>
</tr>
<tr>
<td>Bagging Meta-estimator</td>
<td>81.82</td>
</tr>
<tr>
<td>Ada Boost</td>
<td>81.82</td>
</tr>
<tr>
<td>Gradient Boost</td>
<td>68.18</td>
</tr>
</tbody>
</table>

Figure 4.6: Classification accuracy for different models from three datasets using over-peaked data removal and separated training and test set by labelling the test set manually from the rules’ boundary
Table 4.3: Computational time for different classification models from three datasets using over-peaked data removal and separated training and test set by labelling the test set manually from the rules’ boundary

<table>
<thead>
<tr>
<th>Classification</th>
<th>Computation time (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Temperature</td>
</tr>
<tr>
<td>Gaussian Naïve Bayes</td>
<td>0.008000</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.017000</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.021000</td>
</tr>
<tr>
<td>Extra Tree</td>
<td>0.016000</td>
</tr>
<tr>
<td>Nearest Centroid</td>
<td>0.006000</td>
</tr>
<tr>
<td>Nearest Neighbour</td>
<td>0.008000</td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>0.008000</td>
</tr>
<tr>
<td>Bagging Meta-estimator</td>
<td>0.041000</td>
</tr>
<tr>
<td>Ada Boost</td>
<td>0.213999</td>
</tr>
<tr>
<td>Gradient Boost</td>
<td>0.706000</td>
</tr>
</tbody>
</table>

From the results, when the accuracy is considered, the nearest neighbour and bagging meta-estimator classifiers give the best result for each sensor. When the processing time is considered, the nearest centroid classifier gives the best result for each sensor. This experiment utilises the nearest neighbour which gives the best result for accuracy detection and the computation times is not
much different from the best result. The model is chosen by considering the system reliability which can be applied to reduce an energy usage in an event processing in wireless sensor networks. For example, in term of an accuracy, a parameter for a number of active nodes/high duty cycle nodes can be set as low as possible with an acceptable result to process the same event, the transmission and reception costs in the network can be reduced. In term of computation time, the complexity of the models are concerns which means that the requirement of flash memory is lower than the high complexity models. The usage of flash memory effects to the node lifetime [51]. Moreover, it leads to low energy demand since the processing time for lower complexity is less than the higher complexity that is an energy usage can be reduced. This classifier is performed for the online process in order to detect an outlier/occurrence within a data stream in a room.

4.1.2 Outlier Analysis

Outlier processing in this section describes a detection method for an online processing which utilises the classifier model from the previous section, nearest neighbour. Moreover, the rules which are used for data labelling are performed to classify outlier. The rules include “very low”, “low”, “medium”, “high” and “very high” which are grouped into three classes, “low”, “medium” and “high” for clustering and labelling by grouping “very low” and “low” to the same class and “very high” and “high” to the same class.

The sensor data is processed online and then converted as sliding windows which have a window size 100 (100, 90). The first tuple represents the window size and the second tuple represents the number of non-overlapped window data. Each window is fed into the classifier in order to classify the window data into one of the groups to clarify the environment which is in an abnormal or normal status. The group “medium” is normally a normal situation so an outlier is not processed from here. However, the other groups can have an outlier when it is within the range of “very low” or “very high” from the rules which is created from historical data or in other words, the classified data “low” and “high” from the model can possibly be an outlier if the window is checked from the rules to be “very low” from “low” class or “very high” from “high” class. An event can be
detected from these classes by considering an extremely change of an occurrence such as fire detection. The thresholds for outlier detection are dependent on applications which cannot be fixed and it is better if it can be adapted. Moreover, it is found that the training dataset for the classification model also affect the detection since the environment changes in time. For example, the training dataset in this experiment is collected in summer (June 2014) and the classifier is used in winter (November) which can affect the threshold within the model. In this case the temperature sensor always detected as “very low”. The solution to this problem is to perform adjusted rules for the statistic values which can be the future work. The process of outlier detection is shown in Figure 4.8.

![Figure 4.8: Online processes of outlier detection](image)

When an outlier is detected, it cannot be assumed that an event has occurred in that area since it can be noise, error or false. This step can be a primary process to identify the status of that environment. The collaborative processing can complete further processes in order to detect an event from distributed nodes which can confirm such primary detection as an event and it is added for the future work.
4.2 Summary and Discussion

The work described in chapter 4 can solve the over-fitting problem in classifier evaluation by performing training and test dataset in two different methods, the training dataset is from clustered sensor data with labelling in the previous chapter and the test set is from manually data creation and labelling. The sensor data is analysed, the data distribution is not the same for different sensors and only temperature data can be fitted to the pre-defined rules. These rules are also used for data labelling, test set creation and outlier detection. The dataset which is not in normal distribution cannot utilised these rules because the rules’ boundaries are out of scope. Over-peak data removal is proposed for new statistical values and then the rules are modified to support sensor data which is not in the normal distribution (light and microphone) so an out of scope of rules’ boundaries is solved. The new rules and new testing set are performed for classifier evaluation in terms of detection accuracy and computation time. Nearest neighbour shows the best output for detection accuracy while nearest centroid shows the best computational time. This experiment selected nearest neighbour which gives the best result for accuracy whilst the speed is still high.

After the classifier model is created in offline process, the online process utilises this model in order to detect an outlier in a real situation in a meeting room. The thresholds for rules are redefined since the model is not created from the data which comes from the same period of time when the experiment is performed and the thresholds need to be adjusted for applications. The outliers are processed for collaborative sensing which is added in the future work. It can be considered sensitivity sensors in distributed nodes and then these occurrences are counted from each sensor and node. A threshold is set to define the minimum counting for an event. The event is performed using pre-defined rules from different sensors when there is an outlier over the counting threshold. Finally, the recall of an event is performed for the event confirmation.

This work processes sensor data in order to detect an outlier for event detection. It is completed and analysed as an offline process. They cannot be changed or edited after an installation in an online process. The system should have a rebuilt model or updated model and rules over period of time to enhance the performance when the system is run over months or years.
Chapter 5. Summary and Conclusions

This chapter provides an overview and summary of the thesis for event processing in wireless sensor networks. The contributions are described and highlighted and the approaches to archive the goals are also discussed. This is followed by future work and then conclusions for the key findings and results.

5.1 Summary

This thesis investigates related technologies in statistics, machine learning and data mining for outlier detection and event processing through in wireless sensor networks. The performance is evaluated step by step by choosing the best result from each step. The first target is finding the best results of all components which lead to the optimal clustering including window size and rules. Discrete wavelet transform functions are investigated for a fast detection. These results lead to perform a training dataset of classification, the second stage, for an outlier detection in order to create a suitable classification model for this sensor data. The evaluation metrics of this work are accuracy and computation time. The further investigation is a future work in collaborative sensing to detect an event in real-time (online process) from different sensors and nodes which are in the same room (same environment). The contributions are described in the next section for more details.

5.2 Contributions

The contributions of this thesis can be described in two parts including clustering analysis and classification analysis. The following provides more details on each of the key contributions.
5.2.1 Clustering Analysis

In this work we classify data in three groups using k-means clustering technique and statistic rules to label the data for the groups and then compare cluster performance, homogeneous score and completeness score, for the training data preparation. For this, the sensor data from each node is analysed in the form of overlapping sliding window. The window size affects the cluster performance. Moreover, the rules for labelling the three classes are also considered by comparing between two different statistical keys, a range divided by ten and standard deviation. It is found that using the range gives a better result than standard deviation and it works for normal distribution data. Sensor data does not always create such distribution so the rules are rebuilt to avoid the rules’ boundaries out of the scope of raw sensor data. Another factor, discrete wavelet transform, is performed in data pre-processing in order to reduce data size and noise before clustering. In order to select the discrete wavelet transform function, the functions are compared and evaluated from the detection performance for the fast detection using cosine distance and rules, and coif1 gives the best result.

5.2.2 Classification Analysis

The classification model classifies sensor data into three groups. The classifiers are compared an accuracy performance and it is found that the accuracy results are nearly 100% for every model which is very high. The results have an over-fitting problem because the test set and training set are from the same dataset and it is only grouped into two different groups randomly. In order to solve this problem, the training dataset utilises the data from labelled clusters from the previous stage (clustering analysis) and the test set is generated from a different dataset which is created and labelled manually followed by the rules that are used for cluster labelling. The performance metrics are shown in accuracy and computation time. Nearest neighbour gives the best result in accuracy whilst nearest centroid is the fastest computation time for all sensors. An outlier is processed from sensor reading data in real-time to classify data using the classification model and then is analysed for event processing.
5.3 Discussion and Future Work

This work which is discussed and considered below has investigated and proposed solutions using statistic and machine learning techniques. Some points can be added and improved for the future works.

5.3.1 Discussion

Sensor data is analysed with overlapping sliding windows. It is necessary to define the proper window size however sometimes the size should be adjusted for the system requirements for example early detection or accuracy. The proper size for a good performance of a classifier can be big or small. The smaller size reduces the problem of a delay in data gathering whereas a larger window size can affect the response time.

The cluster techniques which classify data into classes can affect performance and processing time for clustering. The clustering method is an unsupervised technique which normally requires more resources than a supervised technique. This work at first tried to utilise density-based clustering, DBSCAN, but the processing time was very long and took days to process. Moreover, the number of clusters cannot be determined in the beginning, it is shown after the clustering is finished. The parameters need to be adjusted properly in order to have the cluster number as user wants. When the output is not an expectation, the program has to be rerun. The distance-based method is a better option which can process faster with less resource consumption.

A dataset for a classifier creation is also the primary consideration of this work. The time period of the training dataset and a time period using the model should be similar so the model can be performed more effectively. The classifier model and rules are one-time run and cannot update when an environment changes. When the model is utilised over a long period of time, the false rate can be high.

5.3.2 Future Work

The work on event detection technologies described in this thesis opens new possible issues for this research to enhance performances and be more
automatic for different platform. The processing from more than a node or collaborative sensing is a technology can enhance the detection performance which is suitable for distributed system and it will be the next step for this work.

Collaborative sensing is a knowledge processing from shared information by combining the information from a correlated area with disparate sources from a distributed system for decision-making. A process from each node observes an environment and sends a signal to other nodes when there is an occurrence or data request. The signal or qualitative data is collected and shared in order to represent an overall environment from a correlated observed area or its neighbour which can reduce faults from only single node decision-making, micro to macro process. Moreover, an environment situation is processed from environment understanding in distributed nodes in the same area.

In conclusion, the future work will be considered in three different issues, flexible platform, automatic and adaptive and collaboration.

Flexible platform: the technologies in this area enable the creation of a new generation of sensors. The sensor can be any device/mobile device which can effectively share and process data together. To support the large variety of network applications for sensors/mobile devices, a flexible platform needs to be provided.

Automatic and adaptive: automated adaptive rules and model need to be provided for an environment that changes over time. There are three main considerations when designing a system including self-awareness, self-reconfiguration and automation.

Collaboration: the techniques to detect an event from collaborative sensing can enrich context information for a distributed system. An effective system to process ubiquitous data is an autonomous and self-organising system.

5.4 Conclusion

The basic knowledge in WSNs and related techniques are introduced in this document as an Introduction in Chapter 1 and includes the related works, State of the Art analysis (in Chapter 2). An analysis of the state of the art in event
Chapter 5  Summary and Conclusions

detection shows that the researchers have been trying to improve the performance and efficiency of event detection mainly using data mining and machine learning techniques. The objective of this thesis is to extract knowledge from sensor data (noise, errors, anomalies and events) and then clarify/analyse that information to detect an event in WSNs. The environment can be observed and characterised through advanced analysis of the raw data produced by the WSNs like event and anomaly detection.

The event data can be defined by thresholds, rules and/or model(s) which can classify a normal situation and occurrence. The process for this has sub-processes which are important for the next step including historical data analysis (windowing and rules), data pre-processing (DWT), data classification (k-mean clustering and labelling), model creation (classification model) and outlier analysis (threshold and rules). Machine learning creates models automatically from its own system which leads to detection enhancement. The classification and clustering techniques classify data features from a transmitted data. The model creation runs as offline processes and is then utilised to observe an environment in a real-time system, online processes. The observed data and the model is compared to find an outlier in the data stream in order to find an event from such outlier. The distributed system is a character of event detection in sensor nodes so the collaborative event processing can enhance the performance from other related nodes to reduce faults from noise and error which will be performed in the future work.
Bibliography


