Western Graduate&PostdoctoralStudies

Western University

Scholarship@Western

Electronic Thesis and Dissertation Repository

8-26-2016 12:00 AM

Collective Contextual Anomaly Detection for Building Energy Consumption

Daniel Berhane Araya
The University of Western Ontario

Supervisor
Dr. Miriam A.M. Capretz
The University of Western Ontario Joint Supervisor
Dr. Girma Bitsuamlak
The University of Western Ontario

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Engineering Science
© Daniel Berhane Araya 2016

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Computer Sciences Commons, and the Other Electrical and Computer Engineering Commons

Recommended Citation
Berhane Araya, Daniel, "Collective Contextual Anomaly Detection for Building Energy Consumption" (2016). *Electronic Thesis and Dissertation Repository*. 4027.
https://ir.lib.uwo.ca/etd/4027

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

Commercial and residential buildings are responsible for a substantial portion of total global energy consumption and as a result make a significant contribution to global carbon emissions. Hence, energy-saving goals that target buildings can have a major impact in reducing environmental damage. During building operation, a significant amount of energy is wasted due to equipment and human-related faults. To reduce waste, today's smart buildings monitor energy usage with the aim of identifying abnormal consumption behaviour and notifying the building manager to implement appropriate energy-saving procedures. To this end, this research proposes the *ensemble anomaly detection* (EAD) framework. The EAD is a generic framework that combines several anomaly detection classifiers using majority voting. This anomaly detection classifiers are formed using existing machine learning algorithm. It is assumed that each anomaly classifier has equal weight. More importantly, to ensure diversity of anomaly classifiers, the EAD is implemented by combining pattern-based and prediction-based anomaly classifiers. For this reason, this research also proposes a new pattern-based anomaly classifier, the *collective contextual anomaly detection using sliding window* (CCAD-SW) framework. The CCAD-SW, which is also a machine leaning-based framework that identifies anomalous consumption patterns using overlapping sliding windows. The EAD framework combines the CCAD-SW, which is implemented using autoencoder, with two prediction-based anomaly classifiers that are implemented using the support vector regression and random forest machine-learning algorithms. In addition, it determines an ensemble threshold that yields an anomaly classifier with optimal anomaly detection capability and false positive minimization. Results show that the EAD performs better than the individual anomaly detection classifiers. In the EAD framework, the optimal ensemble anomaly classifier is not attained by combining the individual learners at their respective optimal performance levels. Instead, an ensemble threshold combination that yields the optimal anomaly classifier was identified by searching through the ensemble threshold space. The research was evaluated using real-world data provided by Powersmiths, located in Brampton, Ontario, Canada.

Keywords: Anomaly detection, ensemble learning, autoencoder, support vector regression, random forest, building energy consumption

Acknowledgments

I would like to begin by expressing my sincere appreciation to my supervisor, Dr. Miriam Capretz. Dr. Capretz has inspired and supported me during the last two years. I would like to thank Dr. Capretz for giving me the opportunity to study under her; for providing me with continuous mentoring; and most importantly, for having faith in me throughout the ongoing challenges of research. This work would not have been possible without her continued support, time and effort. I am who I am now because of Dr. Capretz's expertise.

My co-supervisor, Dr. Girma Bitsuamlak has provided me with continuous support; ongoing motivation and unlimited patience. I would not have had the opportunity to study Software Engineering at Western University if it was not for him.

I would also like to extend my heartfelt appreciation to my wonderful research team for helping me throughout the course of my research: Sara Abdelkader; Wander Queiroz, Mauro Ribeiro; Dennis Bachmann, and Alexandra L'Heureux. Special thanks to Dr. Katarina Grolinger, Dr. Hany ElYamany and Dr. Wilson Higashino who have tirelessly and patiently spent hours guiding and mentoring me. As challenging as research can be at times, they made it fun! I would also like to thank members of industry I worked with, especially Powersmiths, for their support and constructive discussions regarding the application of research for real-world problems.

I would not be here without the love and support of my family who have been my rock since childhood. I would like to thank my mother, Dehab, for her unconditional love and support. My sister, Tirhas and brother-in-law Temesghen who greeted me upon my arrival to Canada; and did everything in his power to ensure I had a safe and conducive environment to pursue my academic goals. I love you all from the bottom of my heart.

With heartfelt gratitude and appreciation to all my friends and neighbors. In particular, to my Canadian family, Russ and Kathy, who have adopted me as their own. I am blessed to be a part of their family.

Last but not least, I would like to thank all my teachers since childhood who have influenced my life through their teachings.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Anomaly detection refers to the process of identifying abnormal behaviour that does not conform to expected patterns. Identifying anomalies has application in wide-ranging domains including intrusion detection, system health monitoring, building energy consumption monitoring, and others. Anomaly detection takes advantage of labelled or unlabelled datasets to classify data instances as either anomalous or not. The term "labelled dataset" refers to the existence of a label that identifies each data occurrence as either anomalous or not. The output of an anomaly detection system can also be a label or score. A score provides a measure or degree of how anomalous an observation is considered to be.

Anomalies can be broadly categorized as *point*, *contextual*, or *collective anomalies* [4]. *Point anomalies* refer to the occurrence of a value that is considered anomalous compared to the rest of the data. For instance, an hourly heating, ventilating, and air conditioning (HVAC) consumption data point might be anomalous compared to previous recorded hourly values. *Contextual anomalies* take contextual or behavioural attributes into account to identify anomalies. For instance, an hourly HVAC consumption data point may be anomalous in winter, but not in summer. *Collective anomalies* occur when a set of related data instances is anomalous compared to the rest of the data. Individually, these values may not be anomalous, but collectively they represent an anomalous behaviour. For instance, individually, a facility's lighting energy consumption values may be normal as compared to previous recorded values. However, if a set of these values is considered over a specific time window, it may represent a collective anomaly.

Most existing studies focus on *point* and *contextual anomalies*, with fewer studies considering *collective anomalies*. One of the problems of standard collective anomaly detection techniques is that they show little concern for the context of the collective anomaly under consideration. Individually, the values of a collective data point may not be anomalous, but collectively, and more importantly when viewed in a temporal, spatial, or dimensional context, they may represent anomalous behaviour. For example, a collection of heating energy consumption data from a school recorded every five minutes for one hour may be anomalous in the summer (when schools are closed), but not in fall (when schools are running).

1.1 Motivation

An important application of anomaly detection is in the energy and buildings domain. According to the United Nations Environment Programme [5], buildings are the largest contributor to global carbon emissions, accounting for about 40% of global energy consumption and 30% of CO_2 emissions. Moreover, the World Energy Outlook [6] report states that compared to the transportation and industry sectors, buildings have the highest untapped energy efficiency potential. Hence, energy efficiency targets aimed at the building sector can have a significant impact on achieving a green future.

During operation, a significant amount of building energy is wasted due to equipment failure and human-related faults. One approach to energy efficiency is to monitor building energy usage with the aim of identifying abnormal or anomalous consumption behaviour and subsequently taking appropriate energy-saving measures. In recent decades, with the proliferation of sensor devices, modern buildings have been equipped with an increasing number of sensors and smart meters. By analyzing data from these devices, normal consumption profiles can be identified. Subsequently, if consumption patterns that do not conform to the normal profiles are detected, the building manager is notified, and appropriate energy-saving measures are taken. More importantly, for safety-critical building services such as gas consumption, early detection and notification of anomalous behaviour (e.g., gas leakage) can help prevent potentially life-threatening disasters.

The volume and speed at which sensor data are generated makes it challenging to label anomalous consumption patterns. One motivation of this work is to design a framework that relies on unsupervised learning using unlabelled data, to capture the prevalent historical consumption patterns. By assuming that historical data are predominantly normal, the framework builds a model representing normal consumption behavior, after which this model is used to identify whether or not new patterns are anomalous.

Feature generation and selection, which refers to the process of generating and selecting contextual and non contextual features for model building, also presents another challenge. More specifically, defining features to capture anomalous patterns is a difficult task. One motivation for this work is to use feature generation techniques that rearrange consumption data using overlapping sliding windows. Each sliding window contains a consumption pattern, and each data instance is represented by a pattern and its associated generated features. By rearranging the dataset in this manner, the algorithm used can learn to identify normal consumption patterns.

Moreover, to address the performance issues associated with large number of features, the framework uses autoencoders. Autoencoder perform non-linear dimensionality reduction providing computational efficiency [7] and better classification accuracy [8] than other dimensionality reduction techniques such as PCA or Kernel PCA [9].

Another challenging anomaly detection task in this domain is that a single learning algorithm may not capture enough information from the training dataset available. For instance, a single learner may be able to identify only certain aspects of anomalous behaviours. Another motivation of this work is to use several learners which are based on diverse learning techniques and to combine them using majority voting to create an ensemble anomaly classifier with a better-rounded response to different aspects of anomalous patterns.

Most existing anomaly detection techniques [10] [11] determine whether or not a consumption value is anomalous compared to historical consumption data and show little concern for the context of the anomaly under consideration. Some studies have considered context in anomaly detection [12] [13], but have been concerned only with point contextual anomalies. Most importantly, these techniques do not address the issue of anomalous collective data contextually. The collective contextual anomaly detection techniques described in this study serve two purposes: short-term and long-term anomaly detection. Short-term anomaly detection refers to identifying anomalies in shorter time frames or profiles (seconds, minutes), whereas long-term anomaly detection involves longer time frames (weekly, monthly, annual). Short-term anomaly detection helps reduce energy waste and utility cost as well as associated environmental impacts. More importantly, for services where abnormal consumption has life-threatening consequences (e.g., gas leakage), it helps avoid potential disasters. Long-term anomaly detection helps facilities plan strategic energy-saving goals and targets. These tasks can range from equipment efficiency analysis to overall facility maintenance planning.

1.2 Thesis Contribution

The work in this thesis involves several contributions; in this section, the main contributions are presented. The thesis contribution as a whole is on the application of machine learning and the first major contribution is a generic ensemble anomaly detection (EAD) framework. The EAD combines several anomaly classifiers using majority voting to capture various aspects of collective contextual anomalies in building energy consumption; the base anomaly classifiers are assumed to have equal weight. The second contribution is the integration of pattern-based and prediction-based anomaly classifiers to create an ensemble anomaly classifier. One of the main advantages of this is the introduction of diversity into the EAD framework, which fosters learning. Each anomaly classifier learns differently and recognizes a certain aspect of the underlying anomalous behaviour, meaning that together the ensemble provides a better-rounded anomaly classification. The third contribution is that the EAD framework identifies a combined or ensemble threshold that yields optimal anomaly detection and minimizes false positives. By assuming that anomaly detection and false warnings have equal weight, the EAD framework searches through the ensemble threshold space to determine optimal anomaly detection performance.

For this purpose, this thesis presents the *collective contextual anomaly detection using sliding windows* (CCAD-SW) framework, which is the fourth contribution of this work. The CCAD-SW is an anomaly detection framework that relies on unsupervised learning and uses overlapping sliding windows to identify contextually anomalies in collective sensor data. The CCAD-SW framework integrates historical sensor data along with generated and contextual features to identify anomalies. One of the important features of this framework is that it addresses the performance issues associated with high-dimensional datasets using non-linear dimensionality reduction techniques. This helps not only to speed up learning, but also to enhance anomaly detection performance.

The feature preparation technique, more specifically, the data rearrangement style used in the CCAD-SW framework, is also another important contribution of this work. One of the challenges in anomaly detection is capturing anomalous patterns, and hence the dataset was rearranged in such a way that the underlying algorithm can learn the normal behaviour of patterns instead of single values. The CCAD-SW framework is flexible and can adapt to the requirements of the anomaly detection domain. This provides an anomaly detection platform that can be tuned to stringent or lenient requirements with regard to sensitivity, specificity, or an optimal overall value of these two metrics. Besides, this framework also adapts to changes in functionality of the facility under consideration.

The last contribution in this work is that by adjusting the sliding window size, the framework can be used both for short-term and long-term anomaly detection. By identifying anomalies in short time frames, the framework helps avoid energy waste as well as potential disasters. Moreover, anomaly detection over long time frames helps achieve long-term energy-saving goals.

1.3 Organization of this Thesis

The rest of this thesis is organized as follows:

- Chapter 2 provides an outline of background concepts associated with collective contextual anomaly detection as well as a literature review of current anomaly detection approaches for sensor data and ensemble-based anomaly detection techniques. This chapter first provides an introduction to the terminology that is used in the rest of the thesis. Next, an overview of the anomaly detection algorithms used in this study is provided. Finally, this chapter presents a review of current studies on anomaly detection in sensor data as well as ensemble-based anomaly detection approaches.

- Chapter 3 describes the components of the collective contextual anomaly detection approach using sliding window (CCAD-SW) framework. Besides providing an overview of the function of each component, this chapter also describes how each component interacts with others. The description is broken down into three sections: the first is data preprocessing, which describes the data cleaning, feature preparation, and normalization aspects. The second is model training and testing, which highlights the training and testing dataset as well as the model training and testing engines. Finally, the last section covers anomaly detection and notification, discussing the threshold determinator and the anomaly classifier and notifier components.

- Chapter 4 describes the components of the ensemble anomaly detection (EAD) framework and is broken down into two broad sections: training and testing. The training section describes the components involved in the training flow and is further divided into five components: the learner model, the unique error value determinator, the anomaly classifier, the ensembler, the ensemble performance evaluator, and optimal ensemble threshold determinator. Finally the testing section is described.

- Chapter 5 describes the implementation and evaluation of the CCAD-SW and EAD frameworks. First, the evaluation of the CCAD-SW is presented. The dataset and algorithms involved are initially described, followed by the experiments and finally a discussion of the experimental results. Similarly, the second section presents an evaluation of the EAD framework and provides a description of the dataset and the algorithms used in the framework. Furthermore, the experiments and a discussion of the results are presented

- Chapter 6 presents the conclusions for this work, as well as discussion on areas of future work for the CCAD-SW and EAD frameworks

Chapter 2

Background and Literature Review

This chapter has two objectives: first, the background concepts and terms of the topics discussed in this thesis will be presented, and second, an overview of existing anomaly detection studies in the building energy domain in general, and more specifically studies that focus on collective contextual anomaly detection, will be provided. Moreover, related ensemble learning-based anomaly detection studies will also be discussed.

2.1 Concept Introduction

In this section the following concepts and terminology will be introduced: **anomaly detection**, **ensemble learning**, **principal component analysis (PCA)**, **autoencoder**, **random forest** and **support vector regression (SVR)**. This section presents the foundations of the concepts behind the work described in this thesis.

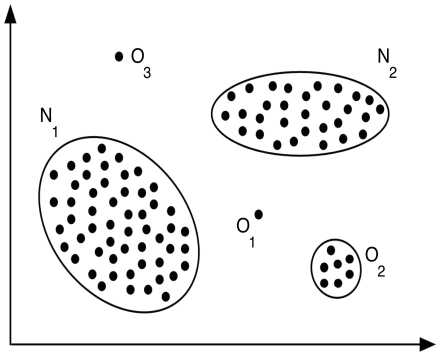
2.1.1 Anomaly Detection

This section discusses various aspects of anomaly detection. In anomaly detection, depending on the domain, several important points must be considered, including *input data*, *type of anomalies*, *availability of data labels*, and *anomaly detection output* [4]. The nature of the *input data* is one of the essential features of any anomaly detection process. How the data are represented and the data types of these representations must be determined. An input refers to a collection of data instances or observations, of which each can be described using a set of features (attributes). Moreover, the features can be of binary, categorical, or continuous type. Binary features are represented by two possible values; categorical features are represented by a categorical number of possible values. For instance, a gender feature may be categorical, with the set of values *male* and *female*. By contrast, continuous features are represented by a continuous range of possible values. For instance, an HVAC sensor consumption reading might be a floating point number [0 5].

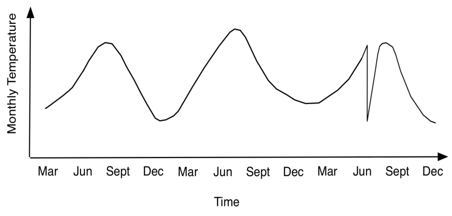
Another important aspect of any anomaly detection technique is the *type of anomaly* under consideration. Depending on the nature of the anomaly, anomalies can be broadly classified as *point*, *contextual* or *collective* anomalies. A description of these terms is given in Table 2.1. The *availability of data labels* is also another feature of an anomaly detection technique and it refers to the availability of labels referring to each observations as either *normal* or *abnormal*. Depending on the availability of data labels, the anomaly detection system can use, *supervised*, *unsupervised*, or *semi supervised* anomaly detection techniques. *Supervised* anomaly detection techniques assume the existence of labelled training data. *Semisupervised Anomaly Detection* techniques typically assume the existence of a small amount of labeled data with a large amount of unlabeled data, whereas *Unsupervised Anomaly Detection* techniques assume that the available training data have no labels. Labelling each observation in a sensor dataset is a difficult and time-consuming process. Moreover, the dynamic nature of anomalies makes it difficult to label these sensor data points. For instance, in the building energy domain, the functionality of a building may change, which in effect changes existing labels. The *anomaly detection outputs* are another important aspect. Normally, anomaly detection outputs are of two types: *scores* which assign a score value to each observation and *labels*, which as already described give each instance is given a label representing *normal* or *anomalous* status.

Term	**Definition**
Point Anomalies	An individual observation is considered anomalous when compared to the rest of the data. A lighting consumption value at a certain time may be higher than previously recorded values. Figure 2.1a illustrates a point anomaly.
Contextual Anomalies	An observation is considered normal in one context, but not in another. As illustrated in Fig. 2.1b, a specifically very low temperature reading might be normal in winter, but not in summer.
Collective Anomalies	A collection of related observations is anomalous when compared to the rest of the data. An hourly profile or pattern of HVAC consumption may be anomalous compared to other hourly patterns. A collective anomaly is illustrated in Fig. 2.1c.

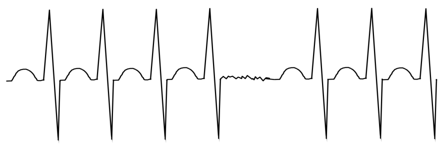
Table 2.1: Types of Anomalies Definitions



(a) Point Anomaly: Represented by O1, O2, and O; these points are outside the concentrated clusters N1 and N2.



(b) Contextual Anomaly: Represented by the low value of temperature value in June, which is abnormal for the month



(c) Collective Anomaly: Represented by the horizontal pattern half way along the graph. This pattern is anomalous when compared to previous normal patterns

Figure 2.1: Types of Anomalies

2.1.2 Anomaly Detection Metrics

This section presents the concepts of the anomaly detection metrics used to analyze the frameworks proposed.

The metrics used to analyze the anomaly detection frameworks proposed in this research are the *sensitivity* and *specificity* [14], which are statistical measures of the performance of binary classification tests. The definition of these terms are provided in Table 2.2. The sensitivity and specificity of an anomaly classifier are evaluated using (2.1) and (2.2) respectively.

$$\mathit{Sensitivity} = \frac{\mathrm{TP}}{\mathrm{P}} \tag{2.1}$$

$$\mathit{Specificity} = \frac{\mathrm{TN}}{\mathrm{N}} \tag{2.2}$$

where True positive (TP) is the number of anomalous consumption patterns that are correctly identified as anomalous, True negative (TN) is the number of normal consumption patterns that are correctly identified as normal, P is the total number of positive instances, and N is the total number of negative instances.

In machine learning and data mining studies, the receiver operating characteristics (ROC) curve [15] [1], is widely used to analyse and visualize classifier performance. The ROC curve is a plot in a unit square of the TPR versus FPR. The FPR refers to the rate of false alarm and is given by (1-TNR). Using the ROC curve, the performance of an anomaly detection model for all possible cut-off values can be evaluated, and the threshold value that optimizes specificity as well as sensitivity can be identified. Various threshold determination approaches have been examined [15] [1]. In this research, the rate of anomaly detection and the false alarm rate are assumed to have equal weight. Based on this approach and noting that the point (0,1) on the ROC curve is the ideal point, the shortest distance d from a point on the curve to point (0,1) as shown in Fig. 2.2, can be evaluated using Eq. (2.3) [15]:

$$d^2 = (1 - sensitivity)^2 + (1 - specificity)^2, \tag{2.3}$$

where d is the shortest distance from a point on the ROC curve to the point (0,1). **This distance is used to determine the threshold value that optimizes both the sensitivity and specificity of the framework [16, 17, 18, 19].**

The area under the curve (AUC) is an effective measure of accuracy which determines the overall inherent capacity of an anomaly classifier to differentiate between normal and anomalous data instances. The maximum (AUC = 1), represents a perfect anomaly classifier. Generally, an AUC closer to 1 indicates better anomaly detection performance [1].

The partial area under the curve (pAUC), defined as the area within a range of false positives or true positives [1] is a performance metric that is well suited for comparing classifiers whose ROC curves cross. An anomaly classifier "A" might have better sensitivity than anomaly classifier "B" in a particular specificity range, whereas anomaly classifier "B" might perform

Term	**Definition**
Sensitivity or *true positive rate* (TPR)	Proportion of anomalous instances identified correctly.
Specificity or *true negative rate* (TNR)	Proportion of normal instances identified correctly..
True Positive (TP)	Number of anomalous instances correctly identified as anomalous.
True Negative (TN)	Number of normal instances correctly identified as normal.
False Positive (FP)	Number of normal instances incorrectly identified as anomalous.
False Negative (FN)	Number of anomalous instances incorrectly identified as normal.

Table 2.2: Anomaly Detection Metrics Definition

better than anomaly classifier “A” in another sensitivity range. Hence, instead of using AUC, which gives an overall combined metric, identifying a particular range and using pAUC provides a better comparison measure. By standardizing pAUC, regardless of the partial region defined, the value of pAUC is always 1 for a perfect ROC curve and 0.5 for a random ROC curve. pAUC can be standardized using Eq. (2.4) [20]:

$$pAUC_s = \frac{1}{2}\left(1 + \frac{pAUC - min}{max - min}\right) \tag{2.4}$$

where pAUC is the partial area under the curve for the selected FPR or TPR range, *min* is the pAUC over the same region of the diagonal ROC curve, *max* is the pAUC over the same region of the perfect ROC curve, and $pAUC_s$ is the standardized partial area.

The trapezoid rule is typically used to evaluate the area under a curve by approximating the region under the curve as a trapezoid and calculating its area.

2.2 Algorithms

This section provides an overview of the algorithms that are used in the CCAD-SW and EAD frameworks. Initially, the idea behind ensemble learning will be described. Later on, the concepts of principal component analysis (PCA), autoencoder, random forest and support vector regression machine learning algorithms will be discussed.

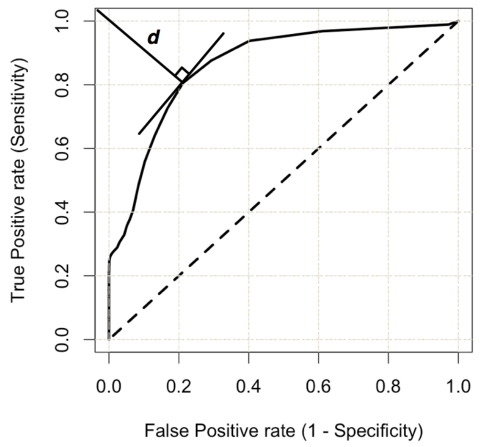


Figure 2.2: ROC: optimal threshold determination [1].

2.2.1 Ensemble Learning

Ensemble learning is a machine learning approach that solves a problem by training multiple learners. Unlike ordinary machine learning techniques in which a single hypothesis is learned from training data, ensemble techniques attempt to build a set of hypotheses and combine them to form a new hypothesis [21]. Previous studies have shown that an ensemble often performs better than the individual learners, also known as *base learners* of the ensemble [22].

Most ensemble techniques rely on a single *base learning algorithm* to produce what are referred to as *homogeneous* base learners. However, some methods use multiple learning algorithms and are referred to as *heterogeneous* learners [21]. The primary objective of ensemble learning is to improve model performance by combining multiple learners.

Normally, ensembles are constructed in two steps. Initially, several base learners are built, and then these learners are combined. Several combination techniques are used. For anomaly classification, *majority voting* [23] is a widely used combination method [21]. In *majority voting*, which is used in this study, the final decision is made based on the agreement of more than half of the base learners.

2.2.2 Principal Component Analysis

Principal component analysis (PCA) is a multivariate statistical technique that is widely used for dimensionality reduction. PCA looks for new orthogonal components (eigenvectors) that explain the largest part of the data variation by providing a measure of the dependency that exists among a set of inter-correlated features [24]. PCA is based on the Eigenvalue Decomposition (EVD) [25] of correlation or covariance matrices or the singular value decomposition (SVD) of real data matrices. The implementation in this thesis is based on SVD. Compared to EVD, SVD is more stable, robust and precise and does not require calculating the correlation or covariance matrix [26].

2.2.3 Autoencoders

An autoencoder [27] is an unsupervised artificial neural network that is trained to reproduce input vectors as output vectors [9]. Fig. 2.3 represents an autoencoder; in this figure, Layer L_1 is the input layer, Layers L_2, L_3 and L_4 are the hidden layers, and Layer L_5 is the output layer. During training, the input dataset $\{x_1, x_2, ..., x_m\}$ is compressed through the three hidden layers into a lower-dimensional latent subspace to reproduce the output $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_m\}$. Assuming that each data sample $x_i \in \mathbb{R}^D$, is represented by a vector of D different variables, the training objective is to construct the outputs by minimizing the reconstruction error in Eq. (2.5).

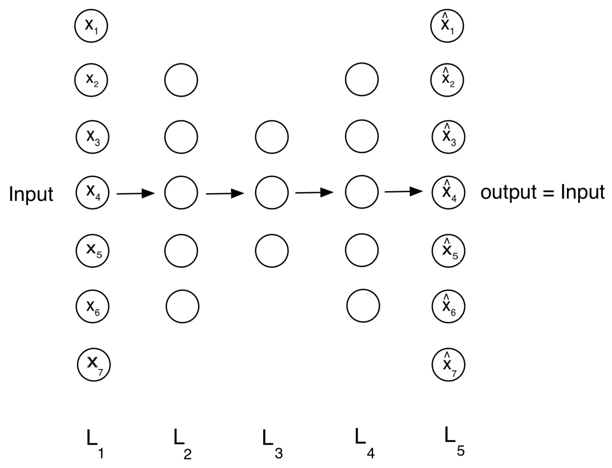

Input
output = Input
x_1 x_2 x_3 x_4 x_5 x_6 x_7
$\hat{x}_1$ $\hat{x}_2$ $\hat{x}_3$ $\hat{x}_4$ $\hat{x}_5$ $\hat{x}_6$ $\hat{x}_7$
L_1 L_2 L_3 L_4 L_5

Figure 2.3: Autoencoder.

$$Err(i) = \sqrt{\sum_{d=1}^{D}(x_d(i) - \hat{x}_d(i))^2} \tag{2.5}$$

The activation of unit k in layer l is given by Eq (2). The sum is calculated over all neurons j in the $(l-1)^{st}$ layer:

$$a_k^{(l)} = f\left(\sum_j W_{kj}^{(l-1)} a_j^{(l-1)} + b_k^{(1)}\right) \tag{2.6}$$

where $\boldsymbol{b}$ and $\boldsymbol{W}$ are the bias and weight parameters respectively. In this study, the hyperbolic tangent is used as the activation function of the autoencoder.

2.2.4 Random Forest

The *random forest* (RF), proposed by Breiman [28] is a widely used ensemble learning method for both classification and regression problems [29] [30]. RF operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes output by individual trees. An RF is composed of an ensemble of B trees $\{T_1(F), ..., T_B(F)\}$, where $F = \{f_1, ..., f_n\}$ is an n-dimensional feature vector. The ensemble produces B outputs $\{\hat{Y}_1 = T_1(F), ..., \hat{Y}_B = T_B(F)\}$ where $\hat{Y}_a$, a=1, ..., B, is the value predicted by the a^{th} tree. The final prediction, $\hat{Y}$, is made by averaging the predicted values of each tree as shown in Fig. 2.4.

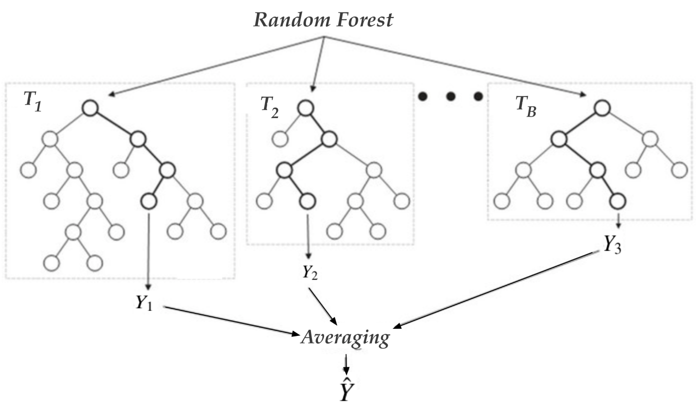


Figure 2.4: Random Forest structure [2].

2.2.5 Support Vector Regression

Support vector machines (SVM) [31, 32, 33, 34] are supervised learning models used for regression and classification purposes. *Support vector regression* (SVR) [35], a version of SVM used for regression, achieves a high degree of generalization, which implies that the model performs very accurately on previously unseen data. The support vectors in SVR are identified from the rest of the training samples by a discriminating loss function that does not penalize residuals less than a tolerance value ε. As a result, for a given hypotheses and ε, the observations constrained to the ε tube bounding the hypothesis, as illustrated in Fig. 2.5 do not affect the predictions.

Given a training dataset $\{(x_1, y_1), ..., (x_N, y_N)\}$, suppose that y is modelled as a function of the input variables x. In SVR, the relationship between x and y is approximated as:

$$y = \omega \cdot \Psi(x) + b, \tag{2.7}$$

where Ψ is a non-linear kernel function that maps from the input space x to a higher-dimensional feature space. The coefficients ω and b are obtained by minimizing the following function:

$$\text{minimize} \qquad \frac{1}{2}||\omega||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \tag{2.8}$$

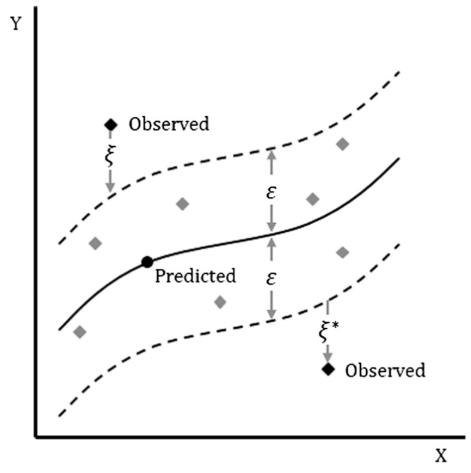


Figure 2.5: Parameters of nonlinear SVR [3].

subject to

$$\begin{aligned} y_i - \omega \cdot x_i - b &\leq \varepsilon + \xi_i \\ \omega \cdot x_i + b - y_i &\leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* &\geq 0 \end{aligned}$$

To achieve good generalization, the weight ω needs to be as flat as possible. The residuals beyond the ε are captured by the terms ξ_i, ξ_i^*, and the cost C is the regularization parameter that determines the penalty for errors greater than ε. This work uses the radial basis function (RBF) because it is a widely used kernel that is efficient to compute. Moreover the kernel has only one parameter that needs to be determined. The RBF kernel is given by:

$$K(x, \acute{x}) = exp(-\gamma ||x - \acute{x}||^2), \tag{2.9}$$

where the kernel parameter γ expresses the influence for each data point.

2.3 Literature Review

This section provides the literature review and is broken down into two sections. The first is anomaly detection for building energy consumption and provides a review of state of the art anomaly detection techniques for building energy consumption presented in academia and the second is ensemble method for anomaly detection which presents current ensemble learning approaches for anomaly detection. These sections are described below.

2.3.1 Anomaly Detection for Building Energy Consumption

Several previous studies used historical building energy data to identify point anomalies [10, 36, 37, 11, 38]. Chou and Telaga [10] proposed a two-stage real-time point anomaly detection system. In their work, consumption value was predicted one-step-ahead using a hybrid neural net ARIMA (auto-regressive integrated moving average) model, and anomalies were identified by comparing whether or not the reading deviated significantly from the predicted value by applying the two-sigma rule.

Janetzko *et al.* [36] outlined an unsupervised anomaly detection system based on a time-weighted prediction that used historical power consumption data to identify point anomalies. Their anomaly detection work described a technique based on a time-weighed prediction and it was compared with a similarity based anomaly detection. The prediction-based anomaly detection gives one anomaly score for each reading while the similarity based anomaly computation gives a score for a daily reading. Wrinch *et al.* [37] detected anomalies in periodic building operations by analyzing electrical demand data using a weekly moving time window in the frequency domain. However, the techniques outlined assumed constant data periodicity which caused many false positives [39].

Hill *et al.* [11] proposed a data-driven modelling approach using one-step-ahead prediction to identify point anomalies. However, their study considered only sequential data and did not take contextual features into account. Considering only historical data to identify anomalies would likely create false positives when contextual information such as season and day of week was included in the anomaly detection process. Bellala *et al.* [38] proposed an unsupervised cluster-based algorithm that identified anomalous points based on a low-dimensional embedding of power data.

In contrast to these studies [10, 36, 37, 11, 38], our research introduces context to the anomaly detection process because a value might be anomalous in one context but not in another. The studies just mentioned considered only point anomalies. However, if a set of values is considered, each value might not be anomalous, but collectively, this set of values might represent anomalous behaviour. Hence, using a sliding window approach, this research identifies contextual anomalies in collective building energy consumption data.

Other studies have considered contextual attributes or behaviours to identify anomalies in a specific context. Arjunan *et al.* [13] proposed a multi-user energy consumption monitoring and anomaly detection technique that used partial contextual information. Besides partially available contextual features, they used the concept of neighbourhood to provide a more relevant context for anomaly detection. Zhang *et al.* [40] used historical data as well as weather and appliance data to compare clustering, entropy, and regression techniques for identifying unusually low energy consumption patterns in a household. The authors focus on ways of using household energy consumption data to identifying vacation days and subsequently remove them so that demand response programs can have accurate data for prediction. Nevertheless, the model presented was static and could not adapt to changes in facility consumption behaviour, for instance, new equipment or a change in building functionality.

Zorita *et al.* [41] presented a methodology that used multivariate techniques to construct a model by using variables that have an influence on energy consumption in a building. The variables considered were climatic data, building construction characteristics and activities performed in the building. The objective of their work was to help building managers of a set of non-residential buildings to compare the energetic performance of their facilities and take appropriate steps. Ploennigs *et al.* [42] presented a diagnostic technique that used a building's hierarchy of sub-meters which provides information on how much energy is consumed by the different building equipments. By analyzing historical data, they identified how abnormal daily energy use is influenced by building equipment and the extent to which exogenous factors affect the energy use of building equipments. The approach used was to compute generalized additive model (GAM) for the main meter and apply an Autoregressive Moving Average (ARMA) model to compute the upper and lower bounds where energy consumption can be considered normal. The method used can determine which meters are abnormal, and whether the anomalies are isolated or persist for a day.

Jiang *et al.* [43] presented a three-stage framework for real-time collective contextual anomaly detection over multiple data streams. These are the *dispatching*, scoring and *alert* stages. The framework is designed to be distributed and can scale to handle large scale data. However, the approach described identifies anomalies in the context of data streams, whereas the proposed CCAD-SW framework is flexible with regard to new contextual features. Peña *et al.* [44] proposed a rule-based system developed using data mining techniques to solve energy inefficiency detection problem in smart buildings. A set of rules was developed using knowledge extracted from sensor data and contextual information. Finally, the results of the rules and energy efficiency indicators were used to construct a decision support system that identifies anomalies.

Capozzoli *et al.* [45] presented an approach to automatically identify anomalies in building energy consumption based on actual recorded data of active electrical power for lighting and total active electrical power of a cluster of buildings. The proposed methodology uses statistical pattern recognition techniques and artificial neural ensembling networks coupled with outliers detection methods for fault detection. The authors mention that by identifying faults in a cluster of buildings, the outlined method can help further optimize the energy consumption by informing occupants of their energy usage and educating them to be proactive in their energy consumption.

Hayes and Capretz [12] outlined a contextual anomaly detection framework for Big sensor Data that used a content anomaly detection algorithm for real-time point anomaly detection. Also, the authors presented a post-processing context-aware anomaly detection algorithm based on sensor profiles. In general, the framework identified sensor data anomalies using a combination of point and contextual anomaly detection approaches.

These studies [13, 40, 41, 43, 42, 44, 45, 12] identified contextual point anomalies. In contrast, our work focuses on a set of consecutive values to identify collective anomalies contextually. Hence, by using a sliding window approach, this study identifies contextual anomalies in collective sensor data. This helps to analyze building consumption profiles contextually over a specific sliding window instead of at a specific point in time. Moreover, by varying the sliding window size, collective contextual anomaly detection can be advantageous in a number of situations. These can range from short-term energy savings and potential disaster prevention objectives to medium- and long-term building energy profile analyses which can be useful in planning long-term energy-saving targets. In addition, this research identifies an anomaly detection framework that optimizes both anomaly detection (hit rate) and false positive rates.

2.3.2 Ensemble Method for Anomaly Detection

Several studies have focussed on enhancing classification accuracy using an ensemble of classifiers. Some used homogeneous classifiers [46] [47], whereas others used heterogeneous classifiers [48] [49] or a combination of both [50, 51, 52].

Using ensemble methods, Cabrera *et al.* [46] examine the problem of distributed intrusion detection in Mobile Ad-Hoc Networks (MANET). More specifically, the authors used an ensemble obtained by training multiple C4.5 classifiers, to evaluate these classifiers on a MANET network for two types of attacks: Denial-of-Service and Black Hole attacks. The authors described a three-level hierarchical system for data collection, processing and transmission.

Didaci *et al.* [47] proposed a pattern recognition approach to network intrusion detection based on ensemble learning paradigms. The authors categorized feature spaces and trained a neural network with separate features to create several classifiers. Subsequently, these classifiers independently performed attack detection, and their results were later combined to produce the final decision. Folino *et al.* [48] introduced an architecture for a distributed intrusion detection by using ensembles that specialized in detecting particular types of attack. Similarly to our framework, the authors used different algorithms with the same dataset to build different classifiers or models. Zhao *et al.* [49] proposed ensemble methods to enhance the anomaly detection accuracy on unsupervised data using density-based and rank-based algorithms. Besides using these independent learners, the authors also considered sequential methods for ensemble learning in which one detection method is followed by another.

Amozegar and Khorasani [50] proposed an ensemble of dynamic neural network identifiers for Fault Detection and Isolation (FDI) in gas turbine engines. The authors first built three individual dynamic neural-network architectures, then constructed three ensemble-based techniques and compared the performance of these models.

Aburomman and Reaz [51] proposed an ensemble construction method that used particle swarm optimization (PSO)-generated weights to create an ensemble of classifiers for intrusion detection. The authors used Local unimodal sampling (LUS) method as a meta-optimizer to find better behavioral parameters for the PSO and created ensemble classifiers using their proposed approach and the weighted majority algorithm (WMA) approach. Their work used a combination of homogeneous and heterogeneous classifiers.

Shoemaker *et al.* [52] studied an ensemble voting method for anomaly detection in supervised learning using random forests and distance-based outliers partitioning. They demonstrated that this approach provided accuracy results similar to the same methods without partitioning. Moreover, the authors also showed that distance-based outlier and one-class support vector machine partitioning and ensemble methods for semi-supervised learning of anomaly detection perform better compared to non-ensemble techniques.

To the best of our knowledge, no previous work has explored ensemble anomaly detection techniques in the building energy domain. Moreover, in contrast to the studies described above, [46, 47, 48, 49, 50, 51, 52], the ensemble anomaly detection (EAD) framework proposed in this research combines several learners and determines a combined threshold (ensemble threshold) value that yields an ensemble anomaly classifier with optimal anomaly detection performance.

2.4 Summary

In this chapter an overview of the concepts involved in anomaly detection were presented. More specifically, an introduction to the terminology and performance metrics in anomaly detection domain was presented. In addition, an introduction to the machine learning techniques and algorithms that are employed in this research was provided. Finally, current anomaly detection studies in the building energy domain as well as ensemble-based anomaly detection techniques were discussed.

Chapter 3

Collective Contextual Anomaly Detection using Sliding Window (CCAD-SW) framework

This research proposes the CCAD-SW framework illustrated in Fig. 3.1. the framework uses historic sensor data, generated features, and contextual features to identify collective contextual anomalies. Moreover, by using flexible sizes of overlapping sliding windows, the CCAD-SW framework accommodates to both short-term urgent anomaly detection requirements as well as long term building energy consumption profile analysis (monthly, annual etc.) that is aimed at long-term energy saving plans. The components of the CCAD-SW framework are described below.

3.1 Data Preprocessing

The term "sensor data" in this research represents a time-stamped record of consumption data recorded at regular intervals. The dataset need to be processed to suit the learning algorithm used. The following sections describe the data preprocessing steps involved.

3.1.1 Data Cleaning

To mitigate the negative impact of noisy and incomplete data on the performance of the CCAD-SW framework, these data must be removed from the dataset. Depending on the problem domain, noisy data are indicated by values outside the valid range. For instance, in the building energy domain, negative electric consumption values are considered noisy. Incomplete data in this context refers to the existence of missing data within a sliding window data. Moreover, the

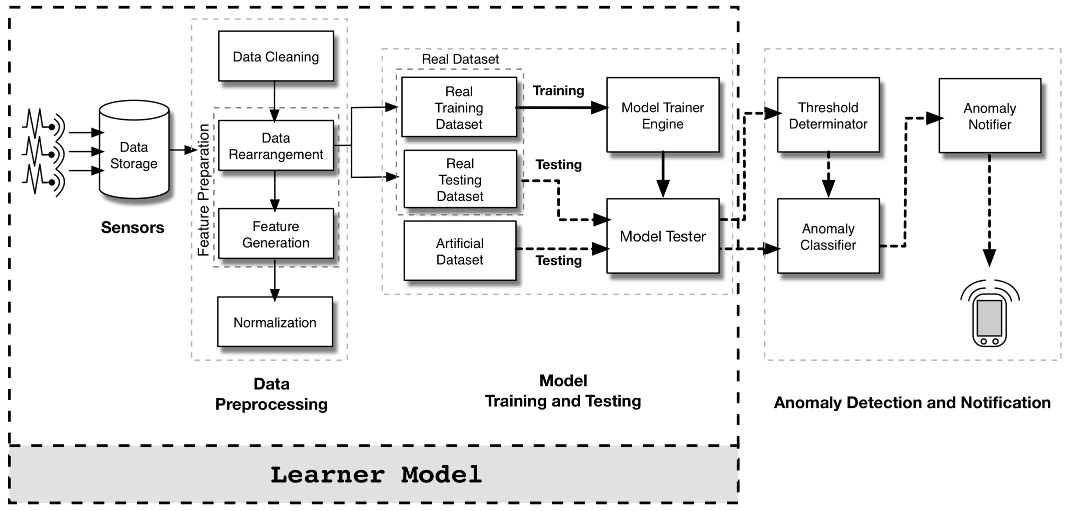


Figure 3.1: Collective Contextual Anomaly Detection using Sliding Window (CCAD-SW) Framework

proposed CCAD-SW framework uses sliding window to identify anomalies; a sliding window in this case is a specific windows size that includes a set of consecutive values. Hence, if the data in a sliding window are incomplete, then that specific input set is removed form the dataset.

3.1.2 Feature Preparation

Given a clean dataset, the feature preparation component focuses on the arrangement and generation of features and involves two sub-steps: ***Data Rearrangement*** and ***Feature Generation***.

- ***Data Rearrangement***: this involves rearranging the sensor data by representing each input instance using sliding window data instead of a single consumption value.

  Table 3.1 shows a sample input dataset of the hourly sliding window data "Sw-a", "Sw-b", and "Sw-c", shown in Fig. 3.2. In the Table 3.1, columns "5", "10", ... , "55", "60" represent an hourly consumption reading recorded every five minutes, and the consumption values in the first, second, and third rows represent the sliding window data "Sw-a",

...	Day of Week	Hour	Minute	5	10	...	55	60	...
...	2	8	45	0.3	0.2	...	0.2	0.2	...
...	2	8	50	0.2	0.2	...	0.2	0.2	...
...	2	8	55	0.2	0.2	...	0.2	0.2	...

Table 3.1: Sample Preprocessed Dataset for CCAD-SW.

“Sw-b”, and “Sw-c” respectively. For instance, “Sw-a” represents an hourly sliding window sensor dataset from 7:45 to 8:45. The next row in the sliding window represents “Sw-b” which is constructed when the data reading at 8:50 becomes available. “Sw-b” represents an hourly sensor dataset from 7:50 to 8:50, and so on. Hence, for instance, for the sample sliding window data shown in Fig. 3.2, collective contextual anomalies are identified every five minutes.

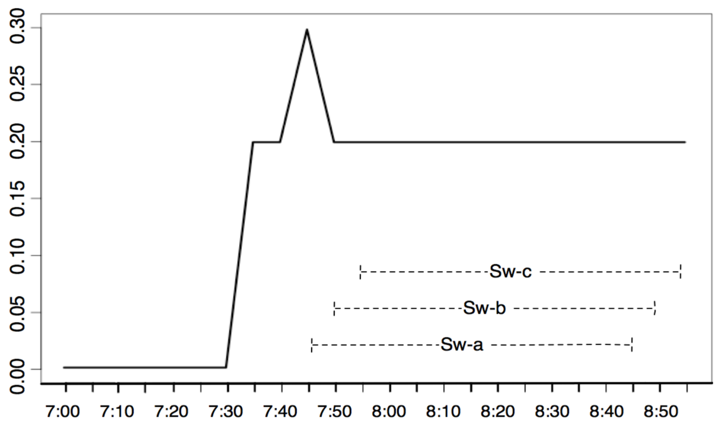


Figure 3.2: Data rearrangement CCAD-SW framework.

- ***Feature Generation***: This component introduces contextual or behavioural features into the CCAD-SW framework. In the building energy consumption domain, the context can be spatial, temporal or weather-related. Moreover, by deriving additional sensor data-generated features such as the mean and median, more insights can be obtained from the sliding window data. The generated features are described in Table 3.2. the temporal contextual features *day of year*, *season*, *month*, *day of week*, *hour of day* are selected because energy consumption exhibits temporal seasonality. To ensure that the CCAD-SW framework does not use features from other sources, weather attributes are not used in this research. The generated features ($\bar{x}$, s, S_n-S_1, $Q1$, $Q2$, $Q3$, and IQR) are selected to explore whether or not these features affect the performance of the CCAD framework. As a measure of central tendency, the mean $\bar{x}$ of the sliding window sensor data provides a measure of the centre of the data. The standard deviation, s, gives an idea of how spread out the data are. The difference between the last and the first elements of the sliding window, (S_n-S_1), shows whether or not the data on both ends of the window are the same. In a dataset that suffers from outliers, the median is a more robust measure of the centre of the data than the mean; hence, the following four features representing the median at different ranges of the sliding window are suggested: the first quartile ($Q1$), second quartile ($Q2$), third quartile ($Q3$) and interquartile range (IQR). A total of 25 features were selected.

Feature	**Description**
Day of Year	1-365/366
Season	1-4
Month	1-12
Day of Week	1-7
Hour of Day	0-23
Minute of the Hour	0-59
$S_j, \{j = 1, ..., n\}$	n - size of the sliding window; S - sensor consumption data
$\bar{x}$	Mean of sensor data values in each window
s	Standard deviation of sensor data values in each window
S_n - S_1	Difference between last and first elements of a sliding window
Q1	First quartile of the sensor data values in each window
Q2	Median of the sensor data values in each window
Q3	Third quartile of the sensor data values in each window
IQR	Interquartile range of the sensor data values in each window

Table 3.2: Features Generated and Domain

3.1.3 Normalization

In a dataset that has features with widely differing scales, the larger values might have more influence than the smaller ones. To give the features equal weight, the dataset was normalized by rescaling the features to lie in the range [0 1] [53] using Eq. (3.1):

$$\acute{x} = \frac{x - min(x)}{max(x) - min(x)} \tag{3.1}$$

where x is the original value and $\acute{x}$ is the normalized value.

3.2 Model training and testing

This section describes the datasets used for model training and testing as well as the model training and testing engines.

3.2.1 Training and Testing Datasets

This section describes the datasets used for model training and testing as well as the training and testing engines. The CCAD-SW framework is based on the assumption that historical sensor data are for the most part normal. Based on this assumption, initially, a subset at the end of this dataset was set aside to assess the specificity of the model. This dataset was not part of model selection (parameter tuning) or model training. Subsequently, the remaining dataset (the "real dataset") was split into real training and real testing datasets. Moreover, an anomalous dataset was artificially generated. These datasets are described below:

- ***Real Dataset*** (D) refers to the historical dataset.
  - ***Real Training Dataset*** (D_{train}) : A subset of the historical dataset that is used to train a model to learn normal consumption behaviour.
  - ***Real Testing Dataset*** (N) : A subset of the historical dataset used to test the specificity of a model.
- ***Artificial Dataset*** (P) : In this research, anomalous data is generated artificially to assess the sensitivity of the model. Artificial anomalous data are generated based on historic sensor data patterns. Consumption patterns can be classified into two types of periods: *high-activity* and *low-activity*. A *high-activity* period has comparatively high energy consumption, whereas a *low-activity* period has either low or zero consumption values. Artificial anomalous data are generated to cover both cases. By plotting the frequency distribution of all the historic consumption data, it is possible to determine the statistically valid range of consumption values that are considered normal. This range is validated by using the 95% confidence interval. An artificial anomalous test dataset for the high-activity period can be generated by fitting an appropriate distribution to the frequency distribution plot and generating random numbers from outside the possible range of consumption values. For the low-activity period, the primary test objective is to determine whether a low-activity period's consumption pattern behaves similarly to an active-period consumption pattern. Hence, for the low-activity period, random consumption values can be generated from the distribution used earlier. But this time, the random values are generated from the range of possible consumption values.

The training method used in this study was a form of *Bootstrap Aggregating* or *bagging* [54], which is a commonly used ensemble modelling technique [55] [56]. *Bagging* is designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. Initially, from the historical dataset, a 10% subset at the end was set aside for final model testing. This dataset was used as a final step to test the specificity of the model and was not part of the model training or validation process. Moreover, the same size of anomalous dataset was generated artificially and used to test the sensitivity of the model.

Let the remaining part of the historical dataset be denoted by D. From D, again, a 10% subset (N) was set aside for model validation. From the remaining 90% dataset, a random 80% (D_{train}) was selected with replacement. The subset D_{train} was used to train a model. Subsequently, this model was tested using (N) as well as the artificially generated anomalous dataset (P). This training and testing process was repeated k times; most published papers suggest a k value between 25 and 50 [56] [54]. After k repeated training and testing cycles, the average of the test results was evaluated, i.e., the average test result for both normal and anomalous test datasets.

Algorithm 1 describes the CCAD-SW illustrated in Fig. 3.1. In the following sections, each component of the figure will be described and the descriptions referred to the corresponding lines in Algorithm 1. The algorithm starts with a loop in line 1 which represents the learning rounds R of the tasks from lines 2 to 9. Inside this loop, initially, bootstrap training samples, D_{train} are generated from the historical dataset D (line 2).

3.2.2 Model Trainer Engine

The *Model Trainer Engine* in this study is a generic component that trains a pattern learning algorithm to reconstruct input data patterns. Autoencoder [27] was used in this research to capture the non-linear and complex pattern that exists between the contextual features and the sliding window of consumption data. Autoencoder provides non-linear dimensionality reduction giving the CCAD-SW framework computational efficiency gains [7] and improved classification accuracy [8] compared to other dimensionality reduction techniques such as PCA or Kernel PCA [9]. The *Model Trainer Engine* can be replaced by other pattern learning techniques. In Algorithm 1, the `ModelTrainer` function trains algorithm “A” to recognize input data patterns consisting of historical sensor data (D_{train}), contextual features (C), and sensor data-generated features (G) (line 3).

3.2.3 Model Tester

Once a model is trained using normal consumption patterns, the `ModelTester` function tests the model using the real testing dataset as well as the artificially generated anomalous dataset. The *Model Tester* component tries to reconstruct the input dataset; the output of this component is a reconstruction error that measures of how close the input data pattern is to the normal data pattern.

Algorithm 1: CCAD-SW framework Algorithm

```
Input  : New Sensor Value ($V$), Real Training Dataset ($D_{train}$), Real Testing Dataset
         ($N$), Artificial Dataset ($P$), Contextual Features ($C$), Generated Features ($G$),
         Number of learning rounds ($R$), Learner Algorithm ($A$)
Output: Notification
/* Function: Learner_Model [lines 1 to 8]                                   */
1  for $i \leftarrow 1$ to $R$ do
       /* Generate bootstrap samples                                         */
2      $D_{train(i)} \leftarrow Bootstrap\ (D)$;
3      normal_model ← ModelTrainer ($A$, $SlidingWindow$, $D_{train(i)}$, $C$, $G$);
4      $err\_neg_i$ ← ModelTester ($normal\_model$, $SlidingWindow$, $N$, $C$, $G$);
5      $err\_pos_i$ ← ModelTester ($normal\_model$, $SlidingWindow$, $P$, $C$, $G$);
6  end
7  $err\_pos = \frac{1}{R}\sum_{i=1}^{R} err\_pos_i$;
8  $err\_neg = \frac{1}{R}\sum_{i=1}^{R} err\_neg_i$;
9  err_value ← ModelTester ($normal\_model$, $SlidingWindow$, $V$, $C$, $G$);
10 thresholdValue ← ThresholdDeterminator($err\_pos$, $err\_neg$);
11 $A_c$ ← AnomalyClassifier($err\_value$, $thresholdValue$);
12 if IsAnomalous($A_c$) then
13     return $Notification$ = true;
14 else
15     return $Notification$ = false;
16 end
```

The `ModelTester` function uses the model trained in the *Model Trainer Engine* to reconstruct new instances of normal historical sensor data as well as artificially generated anomalous data. The reconstruction error array, *err_neg*, which is the test output of the normal test data, as well as the reconstruction error array, *err_pos*, which is the test output of the artificially generated anomalous test data, are determined in lines 4 and 5 respectively.

After R learning rounds, the average of the test outputs of the positive and negative test results are evaluated in (line 7) and (line 8) respectively. In line 9, new sensor data are tested by the `ModelTester` function.

3.3 Anomaly Detection and Notification

The anomaly detection and notification section involves the threshold determination, anomaly detection, and notification steps described below.

3.3.1 Threshold Determinator

The `ThresholdDeterminator` function (line 10) uses the test results of the *Model Tester* component to evaluate a threshold value θ that optimizes the sensitivity and specificity of the CCAD-SW framework. In this component, the density distributions of err_pos and err_neg are determined, and using the TP and TN values of every cut-off error value, the corresponding TPR and TNR ratios are evaluated using Eqs. (2.1) and (2.2) respectively.

The chosen threshold value determines the number of TN and TP captured. A lower threshold value yields a better anomaly detection rate while increasing the false positive rate. The ROC curve was used to determine the threshold that optimized these metrics.

3.3.2 Anomaly Classifier and Anomaly Notifier

The reconstruction error values of new sensor data instances are determined using the trained model. These values are then compared with the threshold value θ, and the `AnomalyClassifier` function (line 11) classifies instances with a reconstruction error value greater than θ as anomalous and instances with an error value less than θ as normal. Anomalous values trigger the `notifier` function to raise an alarm that notifies the building manager, who then performs appropriate energy-saving procedures.

3.4 Summary

In this chapter, the CCAD-SW framework was proposed, and the three major components of the framework i.e., **Data Preprocessing**, **Model Training and Testing**, and **Anomaly Detection and Notification** were discussed. In **Data Preprocessing**, the data cleaning, feature preparation and normalization aspects were explained. In **Model training and testing**, the training and testing datasets used, model trainer engine and model tester components were discussed. Finally, in **Anomaly Detection and Notification**, the threshold determinator, anomaly classifier and anomaly notifier components were elaborated.

Chapter 4

Ensemble Anomaly Detection (EAD) Framework

In this chapter, the Ensemble Anomaly Detection (EAD) framework is proposed. Before delving into the details of the EAD framework, this section briefly explains the motivation and reasoning behind its design.

The generalization ability of an ensemble is usually much stronger than that of a single learner [57]. One of the reasons is that a single learner might not capture enough information from the training data available. For instance, several learners might perform equally well on a given training dataset, but combining these learners might produce a better result. Another reason is that the search processes of the individual learners or base learners might not be perfect. For example, even if a unique best hypothesis exists, it might be difficult to achieve because running the base learners gives suboptimal hypotheses. Thus, ensembles can compensate for such imperfect search processes [57].

Empirical results show that, ensembles tend to have better results when there is a significant diversity among the models [58] [59]. Hence most ensemble approaches tend to promote diversity among the models they combine [60] [61]. One way of introducing model diversity is to use models that are based on different algorithms; another is to use models that are based on the same algorithm, but trained with different subsets of the dataset. To address data shortage issues, this research focusses on the former approach because the latter requires a sizeable dataset.

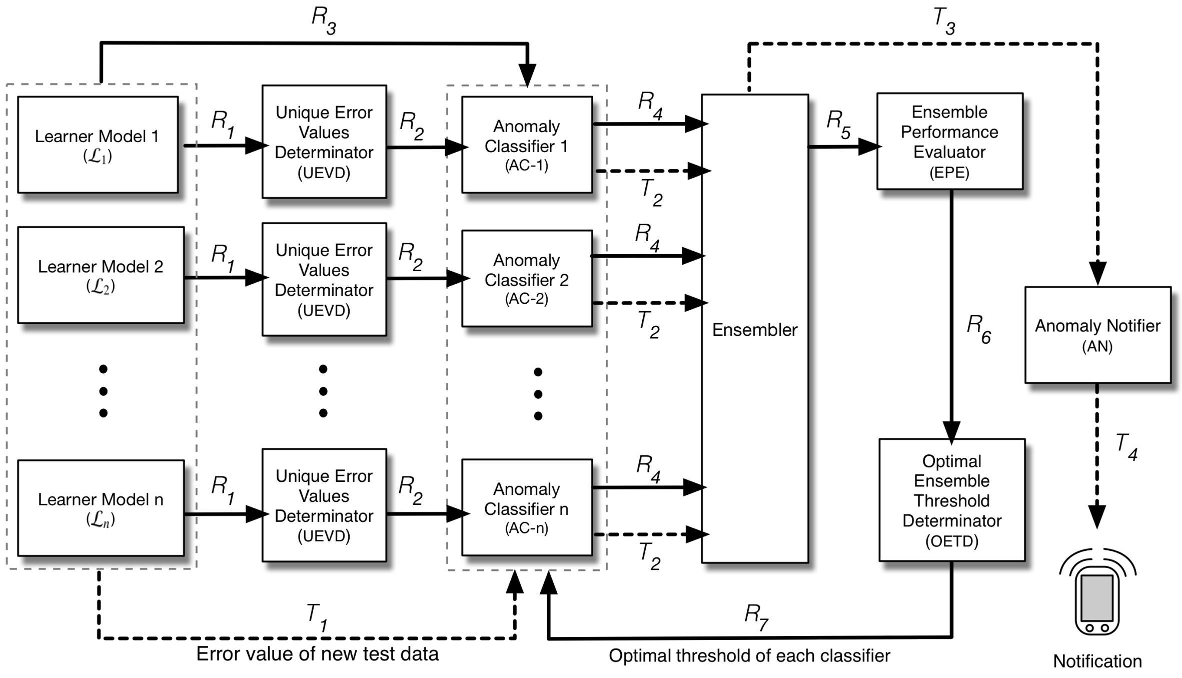
Therefore, this thesis proposes the *ensemble anomaly detection* (EAD) framework shown in Fig. 4.1. The EAD is a generic framework that combines several heterogeneous or homogeneous learners. The framework combines anomaly detection learners that rely on pattern and/or prediction based approaches. Moreover, by evaluating combined threshold (ensemble threshold) values, the EAD framework identifies an ensemble anomaly classifier that yields optimal sensitivity and specificity.

In this study, the prediction-based anomaly classifiers determine whether or not the sum of a sliding windows dataset is anomalous. For instance, from the sliding window dataset illustrated in Fig. 3.2, the CCAD-SW determines whether or not pattern of the sliding window "Sw-a" is anomalous, while the prediction-based learners determine whether or not the sum of the sliding window data "Sw-a" is anomalous. Both of these anomalous classification approaches deal with the same sliding window, but identify anomalous behaviour differently.

The EAD framework, as illustrated in Fig. 4.1 and outlined in Algorithm 2, has training and testing flow paths. These components with their associated lines in Algorithm 2 are described in the following sections.

4.1 Training

The training flow of the EAD framework in Fig. 4.1 is represented by continuous lines and the letter "*R*". The components involved in the training are described in the following sections.



R_3
T_3
Learner Model 1 ($\mathcal{L}_1$)
Learner Model 2 ($\mathcal{L}_2$)
Learner Model n ($\mathcal{L}_n$)
R_1
Unique Error Values Determinator (UEVD)
R_2
Anomaly Classifier 1 (AC-1)
Anomaly Classifier 2 (AC-2)
Anomaly Classifier n (AC-n)
R_4
T_2
Ensembler
R_5
Ensemble Performance Evaluator (EPE)
R_6
Optimal Ensemble Threshold Determinator (OETD)
Anomaly Notifier (AN)
T_4
Notification
T_1
Error value of new test data
R_7
Optimal threshold of each classifier

Figure 4.1: Ensemble Anomaly Detection (EAD) framework

4.1.1 Learner Model (L)

As shown in Fig. 4.1, the EAD framework has several *Learner Models*, of which two types are considered in this study: pattern-based (e.g. CCAD-SW) and prediction-based learner models. The objective of a *Learner Model* is to perform the following tasks: pre-process an input dataset, train a model, test the model using previously unseen normal and anomalous datasets, and finally output these test results.

The use of a *Learner Model* for pattern-based anomaly detection approach has been described in Chapter 3. This section describes the use of a *Learner Model* for prediction-based anomaly detection classifiers.

The *Learner Model*, represented by the largest dashed box in Fig. 3.1, is a generic component of the EAD framework (Fig. 4.1). From Fig. 3.1, the main difference between the application of the *Learner Model* to pattern-based anomaly classifiers and prediction-based anomaly classifiers is, in the *Data Rearrangement*, *Feature Generation*, *Model Trainer Engine*, and *Model Testing* components. In the *Data Rearrangement*, the dataset is reorganized so that the sliding window data shown in Fig. 3.2 are represented by the sum of the consumption data of a sliding window. Table 4.1 shows a sample reorganized dataset with corresponding temporal contextual features for Fig. 3.2. The first, second and third rows of the table represent the input instances for the sliding windows: “Sw-a”, “Sw-b”, and “Sw-c” respectively (Fig. 3.2). Each row contains a single consumption feature, which represents the sum of a sliding window consumption dataset.

In *Feature Generation* these temporal features are also used, but the generated features such as mean and standard deviation are not included because this approach uses a single consumption value as a target variable. In the *Model Trainer Engine*, the underlying algorithm is trained to predict consumption values. In *Model Testing*, consumption is predicted for new data instances and the difference between the actual and predicted consumption values as well as the difference between the anomalous and predicted values is evaluated. These are the outputs of the *Learner Model* for prediction-based anomaly classifiers. The `LearnerModel` function is outlined in Algorithm 1, and called by Algorithm 2 (line 2) (R_1).

Day of Year	Season	Month	Day of week	Hour	Minute	Consumption
142	2	5	2	8	45	2.5
142	2	5	2	8	50	2.4
142	2	5	2	8	55	2.4

Table 4.1: Sample dataset for prediction-based anomaly classifiers

Algorithm 2: Ensemble Anomaly Detection Framework

```
Input : New Sensor Value ($S$), Real Dataset ($D$), Artificial Dataset ($P$), Contextual
        Features ($C$), Learner Models ($L_1$, $L_2$, ... $L_n$), Number of learning rounds ($R$)
Output: Notification
/* Training:Learner_Model and UEVD [lines 1 to 4]                          */
1  for $l \leftarrow 1$ to $n$ do
       /* Function from Algorithm 1                                         */
2      Learner_Model ($D$, $A$, $C$, $G$, $L_l$)
3      $S_l \leftarrow$ uniqueErr ($err\_neg_l$, $err\_pos_l$);
4  end
/* Anomaly Classifier [lines 5 to 10]                                      */
5  for $l \leftarrow 1$ to $n$ do
6      foreach ($\varepsilon \in S_l$) do
7          $H_{neg}L_{l,\varepsilon} \leftarrow$ anomalyClassifier ($err\_neg_l$, $\varepsilon$)
8          $H_{pos}L_{l,\varepsilon} \leftarrow$ anomalyClassifier ($err\_pos_l$, $\varepsilon$)
9      end
10 end
/* Ensembler, EPE and OETD [lines 11 to 19]                                */
11 foreach ($j \in S_1$) do
12     . . .
13     foreach ($z \in S_n$) do
14         $H_{pos}E \leftarrow$ ensembler ($H_{pos}L_{1,j}$, ... , $H_{pos}L_{n,z}$)
15         $H_{neg}E \leftarrow$ ensembler ($H_{neg}L_{1,j}$, ... , $H_{neg}L_{n,z}$)
           /* a 2-D matrix of TPR and FPR of all ensembles                  */
16         $perf \leftarrow$ ensembleMetrics ($H_{pos}E$, $H_{neg}E$)
17     end
18 end
19 $opt\_E_{thresh} \leftarrow$ ensembleOptimizer ($perf$)
/* Testing:MT, AC, Ensembler and AN [lines 20 to 29]                       */
20 for $l \leftarrow 1$ to $n$ do
21     $err\_value_l \leftarrow$ ModelTester ($S$, $C$, $G$)
22     $H_l \leftarrow$ anomalyClassifier ($err\_value_l$, $opt\_E_{thresh}$)
23 end
24 $H_E \leftarrow$ ensembler ($H_1$, $H_2$, ... , $H_n$)
25 if IsAnomalous ($H_E$) then
26     return $Notification$ = true
27 else
28     return $Notification$ = false
29 end
```

Suppose that the EAD framework contains n *Learner Models* denoted by L_l $(l = 1, ..., n)$. As already outlined in Algorithm 1, which uses a single learner, the outputs of a *Learner Model* are the test outputs for normal and anomalous test datasets, denoted by the arrays err_neg and err_pos respectively (Algorithm 1, lines 7 and 8). The EAD framework uses multiple learners, and hence the test results L_l are denoted by err_neg_l and err_pos_l. For a pattern-based learner such as the CCAD-SW, these error values represent the reconstruction error of both normal and anomalous test datasets. For prediction-based learners, these error arrays represent the difference between predicted and actual consumption values. i.e., for real training data, err_neg_l, and for anomalous data, err_pos_l.

4.1.2 Unique Error Values Determinator (UEVD)

Once each L_l has determined the prediction error arrays err_neg_l and err_pos_l, the `uniqueErr` function uses these error arrays to determine a set $S_l = \{\varepsilon_1, \varepsilon_2, ..., \varepsilon_n\}$ that contains unique values of these errors (line 3) (R_2).

4.1.3 Anomaly Classifier (AC)

The objective of the *Anomaly Classifier* is to determine the anomaly class of the outputs of a *Learner Model* L_l by using each of the unique error values $\varepsilon \in S_l$ as threshold values.

The for loop in line 5 runs over all learner models and for each learner model L_l, each unique error element $\varepsilon \in S_l$ (line 6) is used as a threshold value to determine the hypotheses of the err_neg_l, which is $(H_{neg}L_{l,\varepsilon})$ (line 7) and that of the err_pos_l which is $(H_{pos}L_{l,\varepsilon})$ (line 8).

A sample illustration of the output of the *Anomaly Classifier* for err_pos_l, is shown in Fig. 4.2. The 2D matrix in box "1" $(H_{pos}L_{1,j})$ is the hypothesis of err_pos_1 for all unique threshold values $j \in S_1$. Similarly, the 2D matrix in box "n" ($H_{pos}L_{n,z}$) is the hypothesis for all unique threshold values $z \in S_n$. More specifically, each column of the 2D matrices represents the hypothesis of the array err_pos_l determined by using the corresponding unique error value as threshold value. For instance, the first column of the 2D matrix in box "1" represents the hypothesis using threshold value $j = \varepsilon_1$. In the same manner, the second and third and other columns are the hypotheses using threshold values $j=\varepsilon_2$, ε_3, ... ε_p. As a result, the l^{th} *Anomaly Classifier* creates $n(S_l)$ different anomaly classifiers, where $n(S_l)$ is the number of elements of set S_l. For instance the first *Anomaly Classifier* creates p anomaly classifiers, and the last, which is the n^{th}, creates r anomaly classifiers.

The objective of determining all possible classifiers of all learner models is to enable the EAD framework to choose the best combination of threshold values of each *Learner Model* that yields optimal sensitivity and specificity from the entire ensemble.

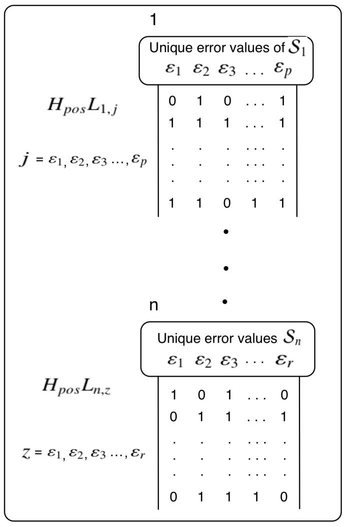

1
Unique error values of S_1
ε_1 ε_2 ε_3 ... ε_p
$H_{pos}L_{1,j}$
$j = \varepsilon_1, \varepsilon_2, \varepsilon_3 ..., \varepsilon_p$
n
Unique error values S_n
ε_1 ε_2 ε_3 ... ε_r
$H_{pos}L_{n,z}$
$z = \varepsilon_1, \varepsilon_2, \varepsilon_3 ..., \varepsilon_r$

Figure 4.2: *Anomaly Classifier* illustration.

4.1.4 Ensembler

The objective of the *Ensembler* is to combine the hypotheses of all the *Anomaly Classifiers* to create a new ensemble anomaly classifier. In this research, by assuming that all learners have equal weight, all possible combinations of all *Anomaly Classifiers* are used to create ensemble anomaly classifiers. The majority vote of the anomaly classifiers is used as the decision or output of the *Ensembler*. By using all combinations of all *Anomaly Classifiers*, the *Ensembler* creates $n(S_1) \times n(S_2) \times ... \times n(S_n)$ different ensemble anomaly classifiers. For instance, one sample ensemble from Fig. 4.2 is an ensemble formed by the majority vote of the hypotheses $(H_{pos}L_{1,3}$, ... , $H_{pos}L_{n,5})$.

The `ensembler` function uses a combination of each *Anomaly Classifier* in each *Learner Model* to combine them and create new ensemble classifier (lines 14 and 15) (R_5). $H_{pos}E$ refers to the anomaly class of an ensemble model for positive test data, whereas $H_{neg}E$ refers to the anomaly class of an ensemble model for a negative test dataset.

4.1.5 Ensemble Performance Evaluator (EPE)

This component determines the anomaly performance of an ensemble classifier. For every ensemble classifier that the `Ensembler` combines, the `ensembleMetrics` function evaluates the performance metrics TPR and TNR of the anomaly classifier using Eqs. (2.1) and (2.2) respectively (line 16) (R_6). By combining all possible anomaly classifiers, the *Ensembler* determines all possible ensemble anomaly classifiers, and the *Ensemble Performance Evaluator* determines the performance of each ensemble.

4.1.6 Optimal Ensemble Threshold Determinator (OETD)

The *Optimal Ensemble Threshold Determinator* (OETD) determines an ensemble threshold opt_E_{thresh} (line 19) (R_7) that optimizes both sensitivity and specificity. The ensemble threshold, opt_E_{thresh} is a set of error values $\{\varepsilon_a, \varepsilon_b, ..., \varepsilon_n\}$ where, $\varepsilon_a \in S_1$, $\varepsilon_b \in S_2$, ..., $\varepsilon_n \in S_n$, such that this combination of error values yields optimal ensemble performance. To identify this ensemble threshold, the ROC curve is plotted using the performance values evaluated in line 16. The ROC plot of the ensemble depends on several learners with different sets of unique error values. As a result, as shown in Fig. 4.3, the ROC curve is not a single curve but a scattered plot in the unit square. The reason is because multiple configurations of the base learner thresholds can yield the same FPR and TPR values.

The ensemble threshold combination that yields the optimal ensemble anomaly classifier is determined using the threshold determination technique described earlier, which assigns equal weight to sensitivity and specificity. The output of the training flow path is a trained model and a set of ensemble threshold values that yield the optimal ensemble anomaly classifier.

4.2 Testing

The testing flow path of the EAD framework in Fig. 4.1 is represented by the dashed lines and the letter "T". During testing, each *Learner Model* L_l initially determines its corresponding

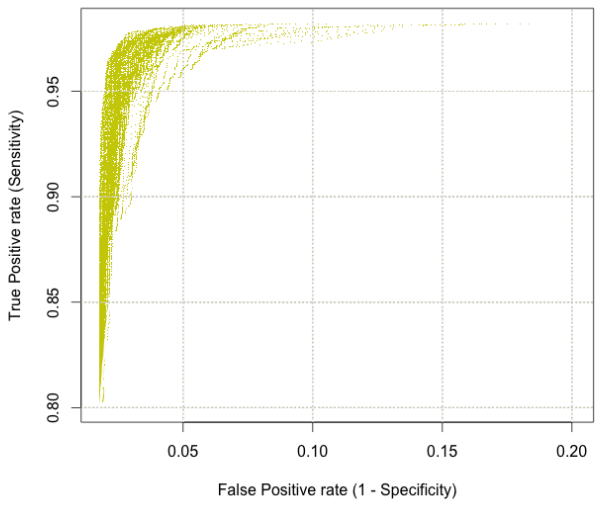


Figure 4.3: Ensemble ROC.

test output array *err_value*$_l$, which is the error measure for a previously unseen dataset (line 21) (T_1). Subsequently, each corresponding anomaly classifier decides about the anomaly class of the same data instance. More importantly, using the optimal threshold values of each learner determined during training by the OETD (line 19) (R_7), the *Anomaly Classifier* determines the anomaly class of the *err_value* (line 22) (T_2). By using majority vote of the decisions of the anomaly classifiers, the `ensembler` function finally decides whether or not an instance is anomalous or not (line 24) (T_3). If the *Anomaly Classifier* has decided that a data instance is anomalous, the `isAnomalous` function triggers the *Anomaly Notification* (line 26)(T_4). The notification can be displayed on dashboard or sent by email, SMS or other interface. Subsequently, the responsible entity, in this case building managers, perform appropriate energy efficiency procedures.

4.3 Summary

In this chapter, the EAD framework was proposed, and the two major flow paths, i.e., **Training** and **Testing** were discussed. In the **Training** flow path, the *Learner Model*, *Unique Error Values Determinator*, *Anomaly Classifier (AC)*, *Ensembler*, *Ensemble Performance Evaluator*, and *Optimal Ensemble Threshold Determinator* components were discussed. Finally in the **Testing** flow path, an overview of the components involved in this path, i.e., *Learner Model*, *Anomaly Classifier* and *Anomaly Notifier* was provided.

Chapter 5

Evaluation and Experiment

In this chapter the implementation and evaluation for the collective contextual anomaly detection using sliding window (CCAD-SW) and ensemble anomaly detection (EAD) frameworks will be presented. Subsequently, the proposed CCAD-SW and EAD frameworks will be evaluated using datasets provided by Powersmiths [62], a company that focusses on producing sensor devices with the aim of creating a sustainable and green future. Powersmiths collects various data from sensor devices, and both of the proposed frameworks were evaluated using HVAC consumption data (kWh) for a school recorded every five minutes from 2013 to 2015. In this work, historic dataset is assumed predominantly normal and based on this dataset, artificial anomaly dataset is generated. Then, the implementation is evaluated with normal as well as artificially generated anomalous datasets. Moreover, this chapter will provide a discussion on how well the CCAD-SW and EAD frameworks identify both anomalous and normal data instances.

This chapter will be organized into two sections. The first section will present the evaluation and experiment of the CCAD-SW framework. Moreover, in this section a comparison is made between two implementations of the framework. The second section will present the evaluation and experiments of the EAD framework. Also, in this section, a comparison of the EAD framework is made with CCAD-SW and two other anomaly classifiers.

5.1 CCAD-SW Evaluation

In this section, the experiments, results and discussion of the CCAD-SW framework is presented.

5.1.1 CCAD-SW Experiments

The dataset was initially preprocessed, subsequently generated and contextual features integrated with it. The final dataset consisted of 337640 samples. Next, a 10% subset at the end of the dataset was set aside to assess the specificity of the CCAD-SW framework. Subsequently two experiments were performed, which are described below:

Experiment 1: The objective of this experiment was to determine the sensitivity and specificity of the CCAD-SW framework using autoencoder. The autoencoder used in this research was based on a deep-learning autoencoder implementation in H2O[63], which is a scalable and fast open-source machine learning platform. The experiment was performed within the R [64] programming environment using an H2O API.

Initially, the autoencoder model was tuned. Various configurations of the regularization parameter (L1), number of epochs, and the activation function as well as both shallow and deep networks were considered. A model that resulted in a minimum stable MSE was finally selected. The parameters selected for the model are given in Table 5.1.

The next step was repeated training and testing; to perform this, from the remaining dataset,

Parameter	**Value**
Hidden Layers	3
Neurons in Hidden Layers	20, 10, 20
L1 (Regularization Parameter)	1E-04
Number of Epochs	400
Activation Function	Hyperbolic Tangent

Table 5.1: Autoencoder Model Parameters

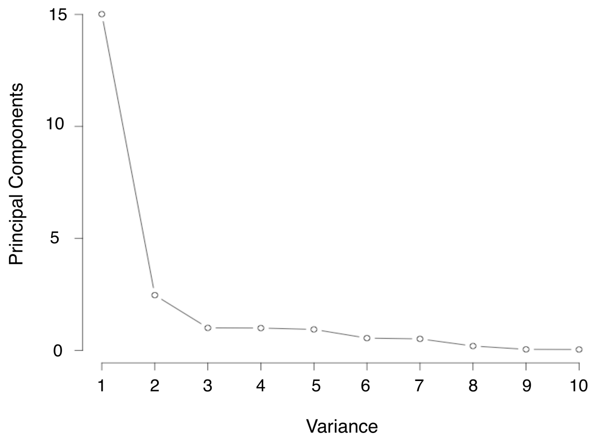


Figure 5.1: CCAD-SW (PCA) scree plot.

a 10% subset was set aside for testing (the normal dataset). Subsequently, from the remaining 90% of the dataset, a random 80% training dataset was selected with replacement. After each training cycle, the model was tested using normal and artificially generated anomalous test datasets. This training and testing cycle was repeated 25 times, and the average values of the reconstruction errors of the normal and anomalous test datasets were evaluated.

Experiment 2: In this experiment, the sensitivity and specificity of the CCAD-SW framework was evaluated using PCA. The dataset that was already used in *Experiment 1* was also used for this experiment. Initially, PCA was used for dimensionality reduction purposes; it was found that the first 10 principal components described 99% of the variance of the dataset, as shown in the *scree plot* in Fig. 5.1. A *scree plot* is a line segment plot that shows the fraction of total variance in the data as explained or represented by each principal component.

During training, the *component loadings* were determined. The *component loadings* are the weights by which each standardized original variable should be multiplied to obtain the component score. These values show how much of the variation in a variable is explained by the component. Subsequently, the principal components that could explain 99% of the variance were selected. During testing, these *component loadings* were used to try to reconstruct previously unseen normal and anomalous test datasets, and the reconstruction error of the normal and anomalous test datasets were determined.

The outputs of both these experiments were the reconstruction errors for positive and negative test datasets. Using these reconstruction errors, the TN and TP density distributions for both experiments were determined. Subsequently, for each experiment, the TPR and FPR = (1-TNR) were evaluated using Eqs. (2.1) and (2.2) respectively. These values were used to plot the ROC curves of the anomaly classifiers. Moreover, assuming that both sensitivity and specificity have equal weight, a threshold value that optimizes these two metrics was determined using Eq. (2.3). Finally, each model was tested using previously unseen normal and anomalous datasets. Subsequently, using the threshold values determined in each experiment, the test outputs of these new datasets were classified as either anomalous or not.

To compare the performance of all these models, the following metrics were used: TPR, FPR, and AUC. In this experiment the ROC curves of the CCAD-SW (autoencoder) and the CCAD-SW (PCA) did not cross the other curves, and hence the pAUC was not considered. To evaluate the AUC, a function *AUC* from the R package that approximates the area under the curve using the trapezoid rule was used. An AUC = 1, which is the maximum value, represents a perfect anomaly classifier, where as an AUC = 0.5 represents a non-discriminant or random classifier.

5.1.2 CCAD-SW results and discussion

The results of *Experiments 1* and *2* are shown in Figs. 5.2a and 5.2b respectively. These are the density distributions of the normalized values of the reconstruction errors. For each error value on the x-axis, the plots show the proportions of TP and TN for the anomaly classifiers. This distribution was referred to as the *TP-TN density distribution* in these experiments. From the figures, the peak of the TN and the peak of the TP for the CCAD-SW (autoencoder) were more separated while for the CCAD-SW(PCA) they overlapped. Intuitively, this indicates that the latter implementation has a lower anomaly detection performance.

The performance of the CCAD-SW implemented using autoencoder and PCA was compared with the performance of models based on the CCAD framework [65]. The CCAD framework used non-overlapping sliding windows, contextual and generated features to identify collective contextual anomalies. Compared to the CCAD, the CCAD-SW framework had better anomaly detection performance and reduced false positives significantly. In addition, by using overlapping sliding windows, the CCAD-SW framework can identify anomalies earlier. The CCAD had two models trained with different features and these were the CCAD-17 and CCAD-26. The CCAD-17 model was trained using seventeen features, twelve of which were consumption data representing an hourly sliding window and five were temporal contextual features. Whereas the CCAD-26 model was trained using twenty-six features, the seventeen features used for the CCAD-17, and nine more generated features. These nine generated features were the ($\bar{x}$, s, S_n-S_1, $Q1$, $Q2$, $Q3$, and IQR) which are already described in Table 3.2 and two more features representing the moving average of the sliding windows, ($\bar{x}_i$-$\bar{x}_{(i-1)}$), ($\bar{x}_{(i+1)}$-$\bar{x}_i$).

Figure 5.3 shows the ROC curve of both CCAD-SW implementations as well as the CCAD-17 and CCAD-26 models. Intuitively, from Fig. 5.3, the CCAD-SW (autoencoder) can be seen to have a larger AUC than any of the other anomaly classifiers. Moreover, the CCAD-SW (PCA) has a curve that almost matches the *line of non-discrimination*, which is the linear diagonal line shown in the figure.

The performance metrics of the four anomaly detection classifiers are shown in Table **??**. They confirm the observations made earlier: CCAD-SW (autoencoder) had the largest AUC followed by CCAD-26, CCAD-17, and lastly CCAD-SW (PCA). If the optimal values of TPR and FPR of each anomaly classifier are compared, with a TPR of 94.5% the CCAD-SW (autoencoder) had the highest anomaly detection rate while still maintaining a FPR of 4.7%, which was the lowest of all the other anomaly detection classifiers. As for the CCAD-SW (PCA), the optimal TPR and FPR measures also showed the same non-discriminant behaviour as indicated by the AUC. Both had values close to 50%. Although the CCAD-SW (PCA) performed slightly better than the random classifier for larger values of FPR (greater than 80%), false positives

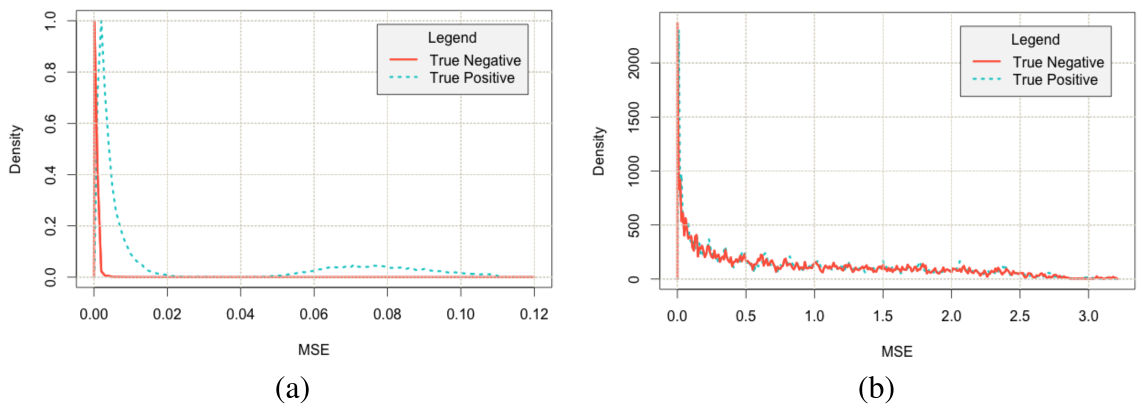


Figure 5.2: TP-TN distribution: (a) CCAD-SW (autoencoder), (b) CCAD-SW (PCA).

in this range would not be acceptable for a workable anomaly detection system. Overall, the experiments done showed that for the dataset used, the CCAD-SW (autoencoder) is well-suited for both lenient and strict FPR requirements.

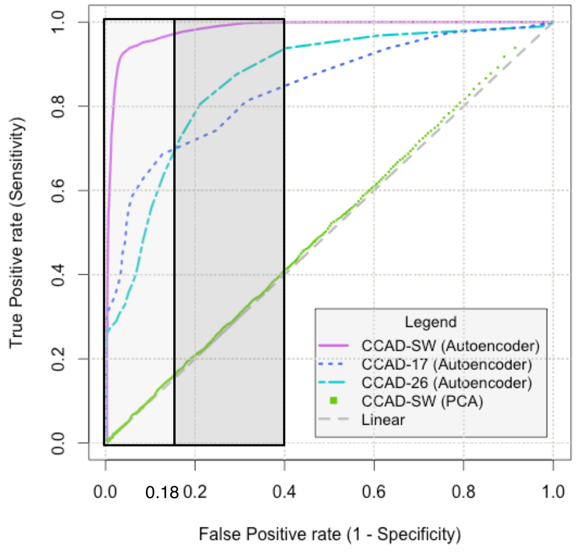


Figure 5.3: ROC of the CCAD-SW and CCAD models.

Model	Threshold	TPR(%)	FPR(%)	AUC	pAUC [1]	
					FPR (0%-6%)	FPR (6%-20%)
CCAD-SW [2]	0.001	94.5	4.7	0.981	0.95	0.95
CCAD-17 [2]	0.07	68.6	12.7	0.842	0.77	0.86
CCAD-26 [2]	0.05	80.2	21.1	0.862	0.72	0.91
CCAD-SW [3]	0.63	52.1	50.4	0.513	0.50	0.51

[1] pAUC standardized [0-1]
[2] Autoencoder
[3] PCA

Table 5.2: CCAD-SW performance comparison

5.2 EAD Evaluation

In this section, an evaluation of the EAD framework implemented by combining the CCAD-SW (autoencoder) framework with two prediction-based anomaly detection classifiers is provided. These two classifiers were implemented using random forest and SVR. A comparison was also made between the anomaly detection performance of the EAD framework and the three anomaly classifiers selected.

The CCAD-SW is a neural network-based learning framework that uses autoencoder. A random forest, is an ensemble learning algorithm that combines the hypotheses of several decision trees, and SVR is a version of support vector machine (SVM) for regression.

The experiment was subdivided into four steps and these are:

- ***CCAD-SW framework based learner model*** : In this step, a CCAD-SW anomaly detection classifier based on autoencoder was implemented and the anomaly class of normal and anomalous test datasets determined.

- ***SVR-based learner model*** : In this step, a prediction-based anomaly detection classifier was implemented using SVR, and the anomaly class of the normal and anomalous test datasets used in the previous step was determined.

- ***Random Forest-based learner model*** : In this step, a prediction-based anomaly detection classifier was implemented using random forest, and the anomaly class of the normal and anomalous test datasets used in the previous steps was determined.

- ***EAD framework anomaly classifier*** : Using majority voting, this step combined the decisions of the three anomaly classifiers.

5.2.1 EAD Experiments

This section provides the details of the experiments implemented to evaluate the EAD framework.

CCAD-SW framework based learner model : This step was already implemented in *Experiment 1*, which was described in the CCAD-SW experimental section. The output of this experiment was the reconstruction error for previously unseen positive and negative test datasets.

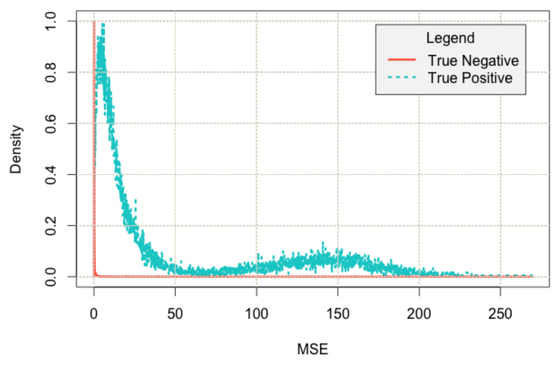
SVR-based learner model : The objective of this step was to implement a prediction-based anomaly detection classifier using SVR. The dataset was initially prepared as described in Section 5.1, and the temporal features mentioned in Section 4.1.2 were introduced into the dataset. The final dataset included a total of 7 features: 6 contextual features and one consumption feature. The same normal and anomalous test datasets used for the experiment described in Section 6.1 were preprocessed and used for final model testing. After preparing the dataset, the parameters of the SVR were tuned by considering various configurations of the parameters C (cost) and γ. By holding one constant and varying the other, a configuration that yielded the minimum MSE was selected; C = 10 and $\gamma = 0.1$.

Next, using the remaining subset of the dataset, the same training and testing technique was applied as described in the CCAD-SW experimental section. The average predicted values of the test runs were determined and using these values, the difference from the actual values were evaluated. The difference between the actual (normal dataset) and the predicted values is the error of the negative test dataset, whereas the difference between the predicted values and the artificially generated anomalous test datasets is the error of the positive test dataset.

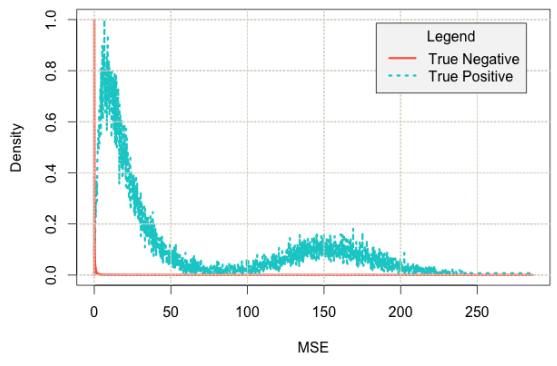
Random Forest-based learner model: The next step was to perform procedures similar to those in the previous, *SVR-based learner model* step, using the random forest algorithm. The same test datasets used in the previous step were also used.

Various random forests with varying tree sizes were considered. For the dataset used, a random forest configuration with 400 trees yielded the minimum MSE value and was selected for the experiment. Using the selected random forest model selected, consumption was predicted and the error values determined as described in the *SVR-based learner model* step.

EAD framework anomaly classifier : The last part of the experiment was to combine the decisions of the anomaly detection classifiers based on the three learner models described so far. The final output of the EAD framework was the anomaly class of the test data as determined by a majority vote of the three anomaly detection classifiers described above.



(a)



(b)

Figure 5.4: TP-TN distribution: (a) Random forest, (b) SVR.

5.2.2 EAD results and discusssion

The TP-TN density distributions of the *Random Forest-based learner model* and *SVR-based learner model* are illustrated in Figs. 5.4a and 5.4b respectively. It is clear that the the peak of the densities of TP and TN are separated, which intuitively shows that the models have a reasonable anomaly detection capacity. The ROC curves of the anomaly classifiers are shown in Fig. 5.5. The ROC plot of the EAD framework is not a single curve because the framework relies on three different learner models. For instance, from the ROC curve of the CCAD-SW (autoencoder) in Fig. 5.3, it can be observed that each FPR value corresponds to one and only one TPR value.

The mean TPR value for each FPR is plotted in the top zoomed figure of Fig. 5.5. Intuitively, the ROC plot shows that the EAD framework outperformed the rest of the individual anomaly classifiers. The ROC curves in figure Fig. 5.5 cross, and hence, for the reasons discussed earlier, the pAUC was also used as a metric to evaluate and compare the anomaly detection classifiers. Moreover, as in the previous experiments, the optimal values of TPR and TNR as well as the AUC were used as performance measuring metrics.

As shown in Fig. 5.5, the ROC curves cross at an FRP of 6%, and hence the pAUC was analyzed for FPR ranges of (0%-6%) and (6%-20%). In this study, because the anomaly clas-

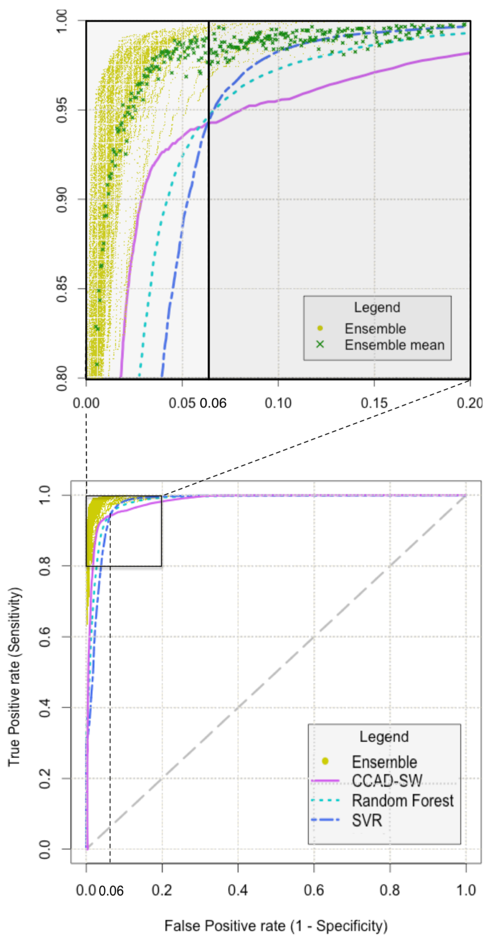


Figure 5.5: ROC of the EAD framework.

sifiers are performed well even for lower false positive rates, only low false positive rates were considered. Moreover, the trend of the curves did not significantly change beyond the ranges considered.

Table 5.3 shows the optimal TPR and FPR values as well as the threshold values that yielded these optimal values. For the EAD framework, the optimal TPR and FPR was achieved at a combined threshold values (ensemble threshold) of CCAD-SW = 0.0032, random forest = 0.3, and SVR = 2. This shows that for the dataset used in this research, optimal ensemble anomaly classifier is not attained by combining the base anomaly classifiers at their respective optimal thresholds which is CCAD-SW=0.001, random forest = 1.7 and SVR = 3.2, but instead at the values CCAD-SW = 0.0032, random forest = 0.3, and SVR = 2.

The table also shows that for FPR range (0%-6%) with a pAUC of 0.95, the EAD framework performed better than any of the base anomaly classifiers. The CCAD-SW (autoencoder) was the second best in this FPR range followed by the random forest based and SVR based anomaly classifiers. Moreover, with a pAUC of 0.97, the EAD framework still outperformed the base learners in the higher FPR range, i.e., (6%-20%). The optimal TPR and FPR values also indicate that the EAD framework outperformed the other classifiers not only in anomaly detection (higher TPR), but also in reducing false positives (lower FPR). Although the SVR-based anomaly classifier had a TPR of 95.7% which is the second best, its has the highest FPR of all the classifiers, 7.10 %. Hence, it may not be suitable for services that require stringent false positive rates. From these experiments, it can be concluded that for all the FPR ranges considered, the EAD is a better anomaly classifier than any of the other base anomaly classifiers.

Model	Threshold	TPR(%)	FPR(%)	pAUC [2]	
				FPR (0%-6%)	FPR (6%-20%)
SVR	3.2	95.7	7.10	0.82	0.965
CCAD-SW [1]	0.001	94.5	4.70	0.89	0.955
Random Forest	1.7	94.9	6.60	0.87	0.960
EAD	[2, 0.0032, 0.3] [3]	98.1	1.98	0.95	0.97

[1] Autoencoder

[2] pAUC standardized [0-1]

[3] Threshold at these values of SVR, CCAD-SW and Random Forest respectively

Table 5.3: EAD performance comparison

5.3 Summary

In this chapter, the evaluation of the frameworks described in Chapters 3 and 4 was presented. Moreover, the implementation details of the experiments as well as the results were discussed. In the CCAD-SW evaluation, the details of the implementation of the CCAD-SW using both autoencoder and PCA was presented. In addition, a comparison of the two implementations was presented. In the EAD evaluation, two more experiments involving prediction-based anomaly classifiers that rely on the machine learning algorithms SVR and random forest were discussed. Finally a comparison of the performance of the ensemble anomaly classifier with the base anomaly classifiers was provided. The results show that the optimal ensemble anomaly classifier is not attained by combining the base anomaly classifiers at their respective optimal threshold. Also, the EAD had better sensitivity and specificity than the base anomaly classifier for all FPR ranges.

Chapter 6

Conclusions and Future Work

This chapter presents a concluding summary based on the contributions of the two proposed frameworks: Collective Contextual Anomaly Detection using Sliding Windows (CCAD-SW) and Ensemble Anomaly Detection (EAD). In addition, a description of possible research on the proposed frameworks will be outlined.

6.1 Conclusions

The work described in this thesis involves a new ensemble-based approach to anomaly detection. To enhance anomaly detection, the ensemble approach combines hypotheses from several learners based on diverse learning techniques. This enables the ensemble to capture enough information from the training data to help learning and hence increase anomaly detection performance. In addition, this thesis presents a pattern-based anomaly detection approach that uses overlapping sliding windows with contextual and generated features to identify collective contextual anomalies in sensor-generated energy consumption data. By varying sliding window size, this approach can be used for both short-term and long-term energy efficiency planning. This thesis has provided the following detailed components of a proof of concept for the CCAD-SW and EAD frameworks:

- Development of an EAD framework that combines hypotheses from diverse anomaly detection techniques in general and that more specifically integrates pattern-based and prediction-based anomaly detection approaches. Moreover, the framework also provides a generic and adaptive learner model that can be used in both anomaly detection approaches.
- The techniques of determining an ensemble threshold, which is a combined base anomaly classifier threshold that yields optimal anomaly detection and minimizes false positives.

- Development of a pattern-based anomaly detection framework (CCAD-SW) that relies on unsupervised learning for contextual identification of anomalous patterns in building energy-consumption data. The framework also addresses performance issues associated with high-dimensional datasets using non-linear dimensionality reduction techniques. In addition, the framework provides a data rearrangement technique that enables the underlying algorithm to learn the behaviour of normal sensor-generated consumption patterns.

In this thesis, the CCAD-SW and EAD frameworks have been evaluated based on detailed implementations. The frameworks were evaluated using HVAC consumption sensor data. Based on the experimental results, the following conclusions can be drawn:

- The CCAD-SW implemented using autoencoder and EAD frameworks successfully identified collective contextual anomalies in artificially generated anomalous data. Moreover, the frameworks could also identify normal patterns in sensor-generated HVAC data. However, the CCAD-SW implemented using PCA behaved as a random anomaly classifier for the dataset and hence is not recommended as a reliable anomaly classifier in this domain.

- The EAD can identify an ensemble threshold that yields an optimal ensemble anomaly classifier. For the dataset used, the optimal ensemble anomaly classifier was not attained by combining the base anomaly classifiers at their respective optimal performance levels. The combined threshold value was rather achieved by searching the threshold space of the anomaly classifiers.

- The EAD framework improved both anomaly detection and false positive rates for the CCAD-SW as well as for two prediction-based anomaly classifiers that relied on support vector regression and random forest algorithms. This makes the EAD framework useful for applications with stringent anomaly detection and false positive requirements.

- The CCAD-SW framework is computationally less intensive than the EAD and hence can be used for services that are less mission-critical, although the EAD is more suitable for critical services.

Anomaly detection plays a significant role in diverse domains of todays systems. More specifically, in buildings, identifying and rectifying abnormal consumption behaviour has far-reaching impact. As world population grows, the need for more food, fresh water, and other resources also increases. This growing demand added to changes in lifestyle has fuelled the demand for more energy. By minimizing energy waste and hence reducing the investment needed for new energy generation and transmission capacity, substantial amounts of money can be saved not only in utility bills, but also by avoiding the repercussions and negative side effects to the environment. Moreover, by identifying long-term building consumption behaviour, facilities can develop strategic efficiency plans that are specifically suited to their buildings. The CCAD-SW and EAD frameworks proposed in this study provide a generic and adaptive platform for identifying collective contextual anomalies in building energy consumption and thereby addressing the energy-related challenges that the world faces today.

6.2 Future Work

This section presents several areas of future work that can be explored:

- The CCAD-SW framework presented in this study considers temporal features to provide context to the anomaly detection process. Future work will consider more diverse features that provide an even broader context. These include features such as occupancy, classroom size, and number of students in a classroom, as well as weather-related features such as humidity and temperature. Although the underlying algorithms used by the frameworks can adapt to changes in building functionality, integrating these contextual features will provide even more valuable learning to the anomaly detection process.

- Another future project is to adapt lambda architecture [66] to identify collective contextual anomalies in real time. A lambda architecture is a data-processing architecture designed to handle massive quantities of data by making use of both batch- and stream-processing techniques. By using robust algorithms to process historical datasets in batch mode and incremental algorithms to create a model using newly generated sensor data, these approaches could subsequently be integrated to identify collective contextual anomalies in real time. This approach to architecture balances latency, throughput, and fault-tolerance using batch processing to provide robust and accurate anomaly detection while simultaneously using real-time stream processing to provide views of on-line data.

- Future work will explore weighted voting to combine base learners. For instance, the decisions of learners with better learning accuracy can be given more weight. More specifically, when using prediction-based anomaly classifiers, learners with better prediction accuracy can have more weight during voting.

- Given large training datasets, future work will consider other ways of introducing diversity to the base learners. One approach is to train each base learner using different subsets of the training dataset. As a result, each anomaly classifier can learn different aspects of dataset behavior, bringing more knowledge into the ensemble anomaly detection process. Another approach is to train the base learners using different features of the dataset.

- The frameworks described earlier require a sizeable historical dataset to create a model that can distinguish between normal and abnormal consumption behaviours and based on the model, classify new sensor data as normal or abnormal. The assumption here is that a facility has been operating for a few years and that during this period sensor-generated consumption data have been recorded. This approach would not be applicable for newly built facilities. An interesting future study in this area would be to use models created for older facilities to evaluate new buildings. To select the right model, introducing contextual features, as described earlier, is important. For new facilities, using models of other facilities located in similar climatic zones and that have similar spatial features and functionality can be explored. For instance, to identify abnormal energy-consumption behaviour in a new school, models already created from another school of similar school size, classroom size, and school hours in the same climatic region can be used.

- In this study, the proposed frameworks were designed with the assumption that historical consumption data is predominantly normal. Future work will explore anomaly detection techniques that consider cases where historical consumption data may have large number of anomalies.

Bibliography

[1] R. Kumar and A. Indrayan, "Receiver operating characteristic (ROC) curve for medical research." *Indian Pediatrics*, vol. 48, no. 17, pp. 277–287, 2011.

[2] C. Nguyen, Y. Wang, and H. N. Nguyen, "Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic," *Journal of Biomedical Science and Engineering*, vol. 6, no. 5, pp. 551–560, 2013.

[3] R. K. Jain, K. M. Smith, P. J. Culligan, and J. E. Taylor, "Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy," *Applied Energy*, vol. 123, pp. 168–178, 2014.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, Jul. 2009.

[5] UNEP, "United Nations Environment Program," http://www.unep.org/sbci/aboutsbci/background.asp., [Accessed 19-Dec-2015].

[6] IEA, "World Energy Outlook." [Online]. Available: http://www.worldenergyoutlook.org/weo2012/. [Accessed 19-Dec-2015].

[7] F. S. Tsai, "Dimensionality reduction techniques for blog visualization," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2766–2773, 2011.

[8] C. Orsenigo and C. Vercellis, "Linear versus nonlinear dimensionality reduction for banks credit rating prediction," *Knowledge-Based Systems*, vol. 47, pp. 14–22, 2013.

[9] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," *MLSDA 2014, 2nd Workshop on Machine Learning for Sensory Data Analysis*, p. 4, 2014.

[10] J.-S. Chou and A. S. Telaga, "Real-time detection of anomalous power consumption," *Renewable and Sustainable Energy Reviews*, vol. 33, pp. 400–411, 2014.

[11] D. J. Hill and B. S. Minsker, “Anomaly detection in streaming environmental sensor data: A data-driven modeling approach,” *Environmental Modelling & Software*, vol. 25, no. 9, pp. 1014–1022, 2010.

[12] M. A. Hayes and M. A. Capretz, “Contextual anomaly detection framework for big sensor data,” *Journal of Big Data*, vol. 2, no. 1, pp. 1–22, 2015.

[13] P. Arjunan, H. D. Khadilkar, T. Ganu, Z. M. Charbiwala, A. Singh, and P. Singh, “Multi-user energy consumption monitoring and anomaly detection with partial context information,” in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pp. 35–44. ACM, 2015.

[14] T. Hill and P. Lewicki, “Statistics methods and applications. statsoft, tulsa, usa,” 2007.

[15] K. Hajian-Tilaki, “Receiver operating characteristic (ROC) curve analysis for medical diagnostic test evaluation.” *Caspian Journal of Internal Medicine*, vol. 4, no. 2, pp. 627–35, 2013.

[16] R. Kumar and A. Indrayan, “Receiver operating characteristic (roc) curve for medical researchers,” *Indian pediatrics*, vol. 48, no. 4, pp. 277–287, 2011.

[17] K. H. Zou, A. J. OMalley, and L. Mauri, “Receiver-operating characteristic analysis for evaluating diagnostic tests and predictive models,” *Circulation*, vol. 115, no. 5, pp. 654–657, 2007.

[18] A. K. Akobeng, “Understanding diagnostic tests 3: receiver operating characteristic curves,” *Acta paediatrica*, vol. 96, no. 5, pp. 644–647, 2007.

[19] M. Greiner, D. Pfeiffer, and R. Smith, “Principles and practical application of the receiver-operating characteristic analysis for diagnostic tests,” *Preventive veterinary medicine*, vol. 45, no. 1, pp. 23–41, 2000.

[20] D. K. McClish, “Analyzing a portion of the roc curve,” *Medical Decision Making*, vol. 9, no. 3, pp. 190–195, 1989.

[21] Z.-H. Zhou, “Ensemble learning,” *Encyclopedia of Biometrics*, pp. 411–416, 2015.

[22] D. Opitz and R. Maclin, “Popular ensemble methods: An empirical study,” *Journal of Artificial Intelligence Research*, pp. 169–198, 1999.

[23] R. Polikar, “Ensemble learning,” ” in *Ensemble machine learning*, pp. 1–34, 2012.

[24] J. C. Lam, K. K. Wan, K. Cheung, and L. Yang, "Principal component analysis of electricity use in office buildings," *Energy and Buildings*, vol. 40, no. 5, pp. 828–836, 2008.

[25] P. Geladi and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Analytica chimica acta*, vol. 185, pp. 1–17, 1986.

[26] P. Praus, "Svd-based principal component analysis of geochemical data," *Central European Journal of Chemistry*, vol. 3, no. 4, pp. 731–741, 2005.

[27] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[28] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[29] K. Ellis, J. Kerr, S. Godbole, G. Lanckriet, D. Wing, and S. Marshall, "A random forest classifier for the prediction of energy expenditure and type of physical activity from wrist and hip accelerometers," *Physiological measurement*, vol. 35, no. 11, pp. 2191–2203, 2014.

[30] A. Kusiak, M. Li, and F. Tang, "Modeling and optimization of hvac energy consumption," *Applied Energy*, vol. 87, no. 10, pp. 3092–3102, 2010.

[31] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[33] V. N. Vapnik and S. Kotz, *Estimation of dependences based on empirical data*. Springer-Verlag New York, 1982, vol. 40.

[34] R. Ž. Jovanović, A. A. Sretenović, and B. D. Živković, "Ensemble of various neural networks for prediction of heating energy consumption," *Energy and Buildings*, vol. 94, pp. 189–199, 2015.

[35] A. Smola and V. Vapnik, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, pp. 155–161, 1997.

[36] H. Janetzko, F. Stoffel, S. Mittelstädt, and D. A. Keim, "Computers and graphics anomaly detection for visual analytics of power consumption data," *Computers and Graphics*, vol. 38, pp. 1–11, 2013.

[37] M. Wrinch, T. H. M. El-Fouly, and S. Wong, “Anomaly detection of building systems using energy demand frequency domain analysis,” *2012 IEEE Power and Energy Society General Meeting*, pp. 1–6, 2012.

[38] G. Bellala, M. Marwah, and M. Arlitt, “Following the electrons: methods for power management in commercial buildings,” *18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’12*, pp. 994–1002, 2012.

[39] R. Fontugne, J. Ortiz, N. Tremblay, P. Borgnat, P. Flandrin, K. Fukuda, D. Culler, and H. Esaki, “Strip, bind, and search : a method for identifying abnormal energy consumption in buildings,” *12th International Conference on Information Processing in Sensor Networks*, pp. 129–140, 2013.

[40] Y. Zhang, W. Chen, and J. Black, “Anomaly detection in premise energy consumption data,” in *Power and Energy Society General Meeting, 2011 IEEE*, pp. 1–8, 2011.

[41] A. L. Zorita, M. A. Fernández-Temprano, L.-A. García-Escudero, and O. Duque-Perez, “A statistical modeling approach to detect anomalies in energetic efficiency of buildings,” *Energy and Buildings*, vol. 110, pp. 377–386, 2016.

[42] J. Ploennigs, B. Chen, A. Schumann, and N. Brady, “Exploiting generalized additive models for diagnosing abnormal energy use in buildings,” *5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pp. 17:1–17:8, 2013.

[43] Y. Jiang, C. Zeng, J. Xu, and T. Li, “Real time contextual collective anomaly detection over multiple data streams,” *SIGKDD Workshop on Outlier Detection and Description under Data Diversity*, pp. 23–30, 2014.

[44] M. Peña, F. Biscarri, J. I. Guerrero, I. Monedero, and C. Léon, “Rule-based system to detect energy efficiency anomalies in smart buildings: A data mining approach,” *Expert Systems with Applications*, vol. 56, pp. 242–255, 2016.

[45] A. Capozzoli, F. Lauro, and I. Khan, “Fault detection analysis using data mining techniques for a cluster of smart office buildings,” *Expert Systems with Applications*, vol. 42, no. 9, pp. 4324–4338, 2015.

[46] J. B. Cabrera, C. Gutiérrez, and R. K. Mehra, “Ensemble methods for anomaly detection and distributed intrusion detection in mobile ad-hoc networks,” *Information Fusion*, vol. 9, no. 1, pp. 96–119, 2008.

[47] L. Didaci, G. Giacinto, and F. Roli, “Ensemble learning for intrusion detection in computer networks,” in *Workshop Machine Learning Methods Applications, Siena, Italy*, 2002.

[48] G. Folino, F. S. Pisani, and P. Sabatino, “A distributed intrusion detection framework based on evolved specialized ensembles of classifiers,” ” in *Applications of Evolutionary Computation*, pp. 315–331, 2016.

[49] Z. Zhao, K. G. Mehrotra, and C. K. Mohan, “Ensemble algorithms for unsupervised anomaly detection,” ” in *Current Approaches in Applied Artificial Intelligence*, pp. 514–525, 2015.

[50] M. Amozegar and K. Khorasani, “An ensemble of dynamic neural network identifiers for fault detection and isolation of gas turbine engines,” *Neural Networks*, vol. 76, pp. 106–121, 2016.

[51] A. A. Aburomman and M. B. I. Reaz, “A novel svm-knn-pso ensemble method for intrusion detection system,” *Applied Soft Computing*, vol. 38, pp. 360–372, 2016.

[52] L. Shoemaker and L. O. Hall, “Anomaly detection using ensembles,” ” in *Multiple Classifier Systems*, pp. 6–15, 2011.

[53] D. Pyle, *Data preparation for data mining*. Morgan Kaufmann, 1999.

[54] L. Breiman, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[55] J. Torres-Sospedra, C. Hernández-Espinosa, and M. Fernández-Redondo, “Using bagging and cross-validation to improve ensembles based on penalty terms,” in *Neural Information Processing*, pp. 588–595. Springer, 2011.

[56] Y. Grandvalet, “Bagging equalizes influence,” *Machine Learning*, vol. 55, no. 3, pp. 251–270, 2004.

[57] T. G. Dietterich, “Ensemble methods in machine learning,” ” in *Multiple classifier systems*, pp. 1–15, 2000.

[58] L. I. Kuncheva and C. J. Whitaker, “Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy,” *Machine learning*, vol. 51, no. 2, pp. 181–207, 2003.

[59] P. S. A. Krogh, “Learning with ensembles: How over-fitting can be useful,” in *Proceedings of the 1995 Conference*, vol. 8, pp. 190–196, 1996.

[60] G. Brown, J. Wyatt, R. Harris, and X. Yao, “Diversity creation methods: a survey and categorisation,” *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[61] J. J. G. Adeva, U. Beresi, and R. Calvo, “Accuracy and diversity in ensembles of text categorisers,” *CLEI Electronic Journal*, vol. 9, no. 1, pp. 1–12, 2005.

[62] Powersmiths, “Powersmiths: Power for the Future,” http://ww2.powersmiths.com/index.php?q=content/powesmiths/about-us.

[63] H2O, “H2O.ai,” http://h2oworld.h2o.ai/#about., [Accessed 19-Dec-2015].

[64] Team, R Core, “R: a language and environment for statistical computing,” http://www.R-project.org/., R Foundation for Statistical Computing, Vienna, Austria, [Accessed 19-Dec-2015].

[65] D. B. Araya, K. Grolinger, H. F. ElYamany, M. A. Capretz, and G. Bitsuamlak, “Collective contextual anomaly detection framework for smart buildings,” in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016.

[66] N. M. J. Warren, *Big Data: Principles and best practices of scalable realtime data systems*. Manning, 2014.

Curriculum Vitae

Name:	Daniel Berhane Araya
Post-Secondary Education and Degrees:	Western University London, ON 2014 - 2016 M.Eng.Sc.
	University of the Witwatersrand Johannesburg, South Africa 2001 - 2002 H.Dip
	University of Asmara Asmara, Eritrea 1993 - 1998 B.Sc.
Honours and Awards:	Western University Graduate Research Scholarship 2014-2016
	University of Asmara Undergraduate Scholarship 1993 - 1998
Related Work Experience:	Teaching Assistant The University of Western Ontario 2014 - 2016
	Database Developer Ping Group 2012 - 2014
	Software Engineer Almaz Computer Technology 2010 - 2012

Publications:

D. B. Araya, K. Grolinger, H. F. ElYamany, M. A. M. Capretz, G. Bitsuamlak, Collective contextual anomaly detection framework for smart buildings, in: International Joint Conference on Neural Networks (IJCNN), IEEE, 2016.

D. B. Araya, K. Grolinger, H. F. ElYamany, M. A. M. Capretz, G. Bitsuamlak, An Ensemble Learning Framework for Anomaly Detection in Building Energy Consumption, Energy and Buildings, 2016 submitted.