Utrecht University

Graduate School of Natural Sciences

MSc Artificial Intelligence

Master's Thesis

Anomaly detection using autoencoders for Ambient Assisted Living

University Supervisor:

Dr. Gerard Vreeswijk

Author:

Riccardo Bellana

Company Supervisor:

Anne van Rossum

4th July 2018

Abstract

One of the main goals of the Ambient Assisted Living field of research is to extend the time of independent living for elderly people through the use of sensors and computational systems. This thesis focuses on building a system which models the power consumption of appliances to build a notion of normality concerning user habits. Once the system learns this notion, it can detect major divergences from it. It is important to detect these major divergences because they can indicate issues in the life of the user. The main research question examines whether autoencoder neural networks are well-suited to detect anomalies in appliance-level power data. For this examination, several autoencoder models with different sets of hyperparameters were trained on appliance-level power data coming from a real-world dataset. Then, the trained models were tested on artificially generated anomalies to evaluate their sensitivity to such data. Autoencoders were shown to perform well with most kinds of anomalies which were deemed relevant in this context. Overall, this thesis analysed important aspects of autoencoders for anomaly detection. Through this analysis, several strengths of the models were found, as well as weaknesses. Moreover, improvements to the current models were proposed to overcome the limitations which emerged from the analysis.

Contents

Chapter 1

Introduction

This chapter provides an introduction to the subject and context of this thesis. First, Section 1.1 illustrates the context that motivated the present research. Then, Section 1.2 provides examples of work that had the same goal as the present research but employed different approaches and tools to achieve it. Next, Section 1.3 presents the main research question of this thesis and the relative sub-questions, followed by the initial hypotheses formulated in Section 1.4. Finally, Section 1.5 outlines the structure of the rest of the document.

1.1 Context

Modern medicine allowed humans around the globe to have a longer life expectancy than ever before. At the same time, in most developed countries the number of newborn decreases year by year. These two concurrent factors shift the past demographic equilibria, leading to an increasingly older population. This shift is well encompassed by the indicator called elderly dependency ratio. This indicator represents the ratio of people eligible to work aged between 16 and 65 years old, to the people outside of this range, namely children and elderly unsuitable for work. Figure 1.1 illustrates the elderly dependency ratio in the European Union from 2006 to 2017 and the projection for the next decades. The data source is Eurostat, the statistical office of the European Union [1, 2]. According to this projection, it appears that we are now at the beginning of a sharp increase in the elderly dependency ratio. This increase will lead in thirty years to a society in which the people of working-age will sustain a body of non-working people half their size.

Elderly people need more medical assistance compared to people of working-age, therefore an ageing population necessarily entails more medical and care assistance. This trend will put under pressure national healthcare systems, and a shortage of hospital beds and caregivers is expected. As an example of this trend, in the UK more than one million elderly are already not receiv-

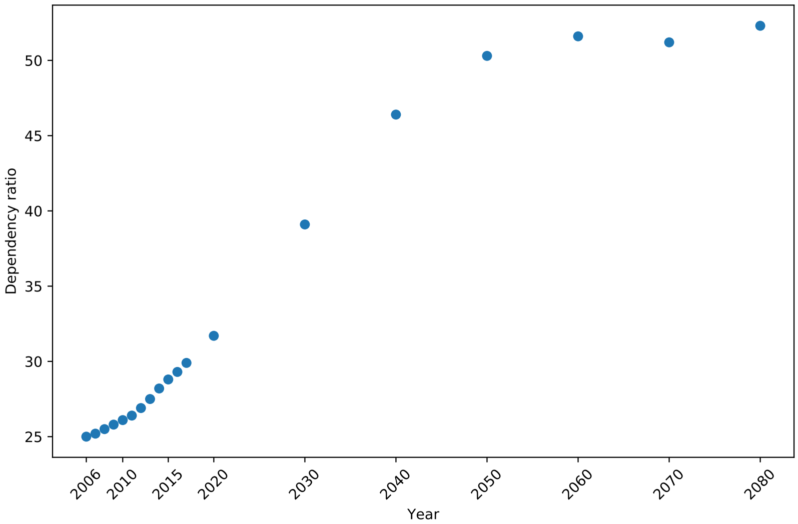

Figure 1.1: Elderly dependency ratio of the European Union.

ing appropriate care [3]. In order to face these problems, and considering most elderly's resolution to retain as much independence as possible, the field of Ambient Assisted Living (AAL) [4] received much attention. The field of AAL has several distinct goals, well presented in the web page of the European Active and Assisted Living programme. The first goal is particularly relevant for this research, and is defined as "extending the time people can live in their preferred environment by increasing their autonomy, self-confidence and mobility" [5]. This goal is pursued through the use of information and communication technologies that monitor elderly people, helping them and their caregivers by supplying services and information. Studies in this context try to automate the task of monitoring elderly and their actions around the house through the installation of dedicated sensors [6]. Such sensors are expensive, as are their deployment and maintenance, making widespread adoption infeasible. On top of the scalability problem, these sensors gather particularly sensitive data, causing a privacy invasion that not many people would accept without question.

On the other hand, smart plugs become increasingly more inexpensive, easy to install and control remotely. On top of these advantages, Noury et al. [7] found that electricity monitoring was considered unobtrusive by the tenants. Moreover, Zhang et al. [8] showed that activities of daily living are correlated with appliance usage patterns. Therefore, monitoring the usage of selected appliances is considered feasible in terms of expense and privacy perception. Considering the above, monitoring appliance-level power usage is a good starting point to detect anomalies in the tenant's activities. A system that has learned the habits of a tenant can serve as a first care solution. Specifically, the system could alert healthcare professionals or family members when it detects a major divergence from normal behaviour. Therefore, this system would help to automate the monitoring task, allowing healthcare systems to scale efficiently. Moreover, this system would give independent elderly higher standards of living by allowing them to enjoy their homes for a longer period of time than otherwise possible.

Crownstone, the company for which this research was done, produces smart plugs and wishes to provide its users with smart services that improve their quality of life. The goal of the present research is to find a way to detect anomalies in appliance-level electrical data. What anomaly means in this context is explained more in depth in Section 3.4, but a general definition together with an introduction to the field of anomaly detection is given in the section below.

1.2 Anomaly detection on appliance-level power data

Chandola et al. [9] give the following broad definition of anomalies: "patterns in data that do not conform to a well-defined notion of normal behaviour". Logically, it follows that anomaly detection refers to the problem of finding those patterns. It is a complex problem that can be tackled with different kinds of techniques, for surveys on this problem see [9–11]. Novelty detection is very much connected to anomaly and outlier detection, but more precisely it is concerned with detecting previously unobserved patterns in the data. For surveys on novelty detection see [12, 13].

In general, anomaly detection on electrical data can be done at different levels of granularity. The current research is interested in appliance-level data because this is the kind that smart plugs are able to record. A lot of research has been done on electrical data coming from office buildings, but the goals were mainly economical, such as, minimizing the electricity consumption and not profiling the tenant's habits. On the other hand, for works that focus on detecting anomalies in the electricity usage patterns of the tenants at household, or building-level see [14–16].

To the extent of my knowledge, only one team of researchers performed anomaly detection on appliance-level electrical data with a focus on the tenant's habits. Their research encompasses Non-Intrusive Load Monitoring (NILM) [17]–a process concerned with finding which appliances determine the current power draw of a house and their respective energy consumption–and anomaly detection on electrical data. The main contributor is José Alcalá and the relevant papers [18–21] were published during his PhD.

In [18], the authors manually extract events from the data and then use a Gaussian Mixture Model to model the normal power usage of an appliance. In order to extract events from the raw power data, the authors set a power threshold for each appliance: every data point that goes beyond that threshold is assigned the label *On*, and every data point below is assigned the label *Off*. However, appliances like washing machines trespass the threshold many times during their cycles. In these cases, only the first time that the threshold is surpassed is considered as a “switch on” event. In the same way, only the last time the values go below the threshold is considered as a “switch off” event. Therefore, data points are ignored according to two conditions: they overcome the threshold and their timestamp value is between the first “switch on” event and the last “switch off” event. Due to this process, some errors may appear but the authors had to consider the trade-off between automatic labelling and accuracy.

In [19], the authors first extract events from the *aggregate* power data using a NILM algorithm they created. Then, they use Cox Processes [22] to model the normal power usage of a single appliance. In contrast with [18], in this work the authors consider only one appliance likely to be used every day, such as a kettle in a British household.

In [21], the authors use the NILM algorithm devised in [19] to extract events and then use the Dempster-Shafer theory [23, 24] to model the normal power usage. Again, the fact that the events are extracted automatically from the raw data might create errors that impact the next step of the analysis.

All the works reported above first extract events from the raw power data, and then try to detect anomalies in this sequence of events. This approach adds a layer of abstraction on top of the raw data, which is both useful and detrimental. On one hand, it is useful because it allows the researchers to frame the problem as an interaction between user and machine, represented by a discrete sequence of events. On the other hand, extracting events from several appliances is not trivial and introduces errors that will trickle down in the anomaly detection task. A technique that is able to deal with raw power data directly would save both a preprocessing step and the risk of introducing errors. Therefore, autoencoders–a type of Neural Network (NN)–were chosen for three main reasons: (1) the ability to deal with the type of data considered, (2) their successful employment in anomaly detection tasks [25–29], and (3) because thanks to recent advances in NN theory, the basic model can be extended to improve its performance.

1.3 Research questions

The main goal of this research is to explore how suitable autoencoders are to detect anomalies in appliance-level power data. The main hyperparameters of the basic model are explored and tested to understand their effect on the performance of the model. This research will serve as a stepping stone for future work, in which the basic model can be extended with more recent innovations to improve the performance of the system. The main research question is reported below, together with three more detailed sub-questions.

Main research question

“Are autoencoders well suited to detect anomalous days in appliance-level electrical data?”
To answer this question, several autoencoder models with different architecture and hyperparameters were trained, and their results explored and evaluated.

Sub-question 1

“What kinds of anomaly are the models able to recognize?”
Data can be anomalous in several ways, therefore it is possible that the models might be performing better on one kind of anomaly, and worse in another. Four kinds of anomalies are devised in Section 3.4, and the trained models are tested against them.

Sub-question 2

“How does performance evolve when the size of the latent representation is increasingly reduced?”
A smaller latent representation implies less flexibility in the reconstruction, and therefore the model should be able to reconstruct only very common patterns. How does this affect the anomaly detection task?

Sub-question 3

“Is there a clear performance improvement when a deep autoencoder is used instead of a shallow autoencoder?”
Deep models store more weights and biases compared to their shallow counterparts. Therefore, they are able to learn a more flexible representation of the training data. How does this affect the anomaly detection task?

1.4 Preliminary hypotheses

For this thesis, three preliminary hypotheses regarding the behaviour of the models were formulated. These hypotheses are based on the understanding of autoencoders as obtained from the reviewed literature and were conceived before the analysis of the results of the experiments. It is useful to report them in order to understand the differences between what was expected and what happened in reality.

Hypothesis 1: the smaller the bottleneck layer, the higher the number of anomalies found.

Hypothesis 2: the anomalies found by models with bigger latent representations are a subset of the anomalies found by a model with a smaller latent representation.

Hypothesis 3: the deep models are able to reach a more flexible learned representation compared to the shallow models. Therefore, the models identify as anomalies only the data points that heavily deviate from the norm.

Hypothesis 1 is motivated by the fact that a small bottleneck layer–in this research the smallest size used is 1/6 of the input size–would lead to a rigid representation of the input data. Because of this rigidity, reconstruction error values would be high on average. It follows that a high number of data points would be identified as anomalous. Hypothesis 2 is motivated by the fact that by shrinking the bottleneck layer size, the model would learn only the increasingly more common features of the data. Therefore, models with smaller bottleneck size will identify as anomalous more data points than models with bigger bottleneck sizes. On the other hand, this means that models with bigger bottleneck size adapts better to the data, while still retaining the same information learned by the model with a smaller bottleneck size. Hypothesis 3 is motivated by the fact that deep models store more weights and biases compared to their shallow counterparts. This gives them more options to replicate the training data. Given this flexibility, most data points will have low reconstruction errors and few will be identified as anomalies.

1.5 Thesis structure

The document is structured as follows. First, Chapter 2 formalizes the problem at hand and explains the mechanisms of autoencoders. Then, in Chapter 3 the data used for the experiments are described. Following this, Chapter 4 illustrates the choices made for training, testing, and evaluating the models. Next, Chapter 5 presents and comments on the results obtained by the models. Chapter 6 compares the present research with the related literature, presenting its limitations, and using them as starting points for future improvements. Finally, Chapter 7 brings the document to a close with a concise conclusion.

Chapter 2

Autoencoders

This chapter illustrates the concepts underlying the usage of autoencoders in the research area of anomaly detection. These concepts will also be the basis for the method proposed in this thesis, which will be introduced in Chapter 4. Section 2.1 gives a more formal statement of the problem at hand compared to how it was previously introduced. Section 2.2 introduces the concept of one-class classification and the class of problems it faces, together with some examples of systems published in literature. Finally, Section 2.3 describes the original autoencoder and several variants developed in recent years.

2.1 Formalism

The problem which the present thesis faces can be formulated inside the frame of reference of the field of anomaly detection. As suggested by other works [18–21], monitoring the usage of several appliances in a household allows to define the normal usage pattern of the appliances of the people living there. Nordhal et al. [15] make the same case for electrical consumption data at a household level. Once the normal usage pattern has been defined, it can be used to detect anomalies as those anomalies might be connected to health issues of the tenant.

The data at hand, described in depth in Chapter 3, have the structure of a set of sequences, where each sequence represents a day and is composed of a number of couples $(timestamp, value)$. The problem is to determine, based on the set of normal sequences, if a previously unseen sequence is to be considered anomalous or not. Formally, given a set S_1 composed of a finite number of time series $[x_1, x_2, \dots, x_n]$, where each element x is composed of a couple $(timestamp, value)$. The time series contained in S_1 describe what is considered to be the usual behaviour of an appliance, and therefore delineate the *normal* class. Additionally, given a set S_2 composed of finite number of time series with the same structure as the one contained in S_1 of *unknown* class, decide for each element of the set S_2 if it belongs to set S_1 based on a metric of similarity of choice. This problem clearly belongs to a class of problems named one-class classification, which is explained more in detail in the next section.

2.2 One-class classification

A way to define one-class classification is to compare it to the more intuitive concept of multi-class classification. Multi-class classification [30] is concerned with assigning an input vector x to one of K discrete classes C_k where $k = 1, \ldots, K$. On the other hand, one-class classification is concerned with discriminating between an object of a known class and objects belonging to *other*, unknown, classes [25, 31]. The main difference between them is that to train a one-class classifier *only* examples from the known class are used, while a multi-class classifier is trained on examples of all classes. Furthermore, as explained by Japkowicz [25], multi-class classifiers draw boundaries *in-between* objects belonging to different classes in the feature space. Differently, one-class classifiers draw boundaries *around* the objects belonging to the known class, with no knowledge about other classes.

Tax [31] proposes a taxonomy of one-class classification strategies, based on the type of the underlying algorithm. Three groups of methods are identified: density estimation methods, boundary methods, and reconstruction methods. Before examining these groups individually, there is an important consideration to make: all these methods have two elements in common, presented below.

1. A distance measure $d(z)$ of an object z to the target class X_{train}, represented by the training set.

2. A threshold θ on the distance measure d. A new object z is classified as belonging to the target class only if $d(z) < \theta_d$.

Density estimation methods like the one proposed by Tarassenko et al. [32] estimate the density of the training data, assuming that the data belong to a certain distribution. Alcalá et al. [18] use a mixture of Gaussians, but also a simple Gaussian or Poisson distributions can be used. Once the distribution is chosen, then it is possible to test new data with several discordancy tests [33]. Tax [31] remarks how these methods work well when a sufficiently high number of data points is available for training, and when a flexible density model, like Parzen density estimation [34] is used.

Boundary estimation methods follow a principle brought forth by the person responsible for the invention and development of Support Vector Machines (SVMs), Vladimir Vapnik. In his words [35]:

> “If you possess a restricted amount of information for solving some problem, try to solve the problem directly and never solve

> a more general problem as an intermediate step. It is possible that the available information is sufficient for a direct solution but is insufficient for solving a more general intermediate problem."

The above quote suggests that calculating directly the boundary which contains the normal class might be easier–i.e. less computationally demanding–than estimating the density of the data *only* to calculate that boundary. Therefore, these methods focus on optimizing a boundary around the target class, and most of them have a strong bias towards a minimal volume solution. Methods like K-center, Nearest neighbour, and Support Vector Data Description belong to this group [31].

Reconstruction methods build a compressed representation of the training data in which noise and redundancies are reduced, while the key information is maintained. For any data point is determined a reconstruction error based on a metric of choice. If this reconstruction error goes beyond a threshold, it means that the input wasn't well represented in the training data and therefore constitutes an outlier. Techniques belonging to this group are Self-Organizing Map (SOM) [36], Principal Component Analysis (PCA) [30], and autoencoders [25]. One advantage of this approach is that the model can improve as new data arrives, updating itself to improve the fit as new data points considered normal become available. The next section analyses autoencoders and their usage in anomaly detection.

2.3 The autoencoder and its extensions

An autoencoder can be easily described as a NN that is trained to reconstruct the data given as input. Being a special case of NN, they are trained with the same techniques: typically gradient descent following gradients computed by backpropagation [37]. The easiest example of an autoencoder is a three-layer NN: the first is the input layer, the second is the hidden layer, and the third is the output layer. In this NN, the hidden layer encodes the input x with a function $h = f(x)$, and the output layer decodes the encoded input with a reconstruction function $\hat{x} = g(h)$, where $\hat{x}$ is the reconstructed input x. The autoencoder minimises a loss function, in the case of Mean Squared Error (MSE):

$$L(x, \hat{x}) = \frac{1}{n} \sum_{i=1}^{n} (x - \hat{x})^2 \tag{2.1}$$

This is the reason autoencoders were placed in the reconstruction methods in the taxonomy presented in Section 2.2. Finding a function $g(f(x)) = x$ seems trivial, and here is where things get more interesting; autoencoders are designed to be unable to copy their input perfectly [37].

Autoencoders are constrained in the sense that they are allowed to only copy input similar to the training data. An explicit way to do so, and also the first to be used in literature [38], is to choose a number of nodes for the hidden layer that is smaller than the number of nodes in the input layer. This way, the autoencoder learns a compressed representation of the input data and is only able to optimally reconstruct input which was seen frequently in the training data. In other words, useful properties of the data are learned because the model needs to minimize the reconstruction error while retaining less information than the original data.

Rumelhart et al. [38] made the first explicit mention of an autoencoder in 1985, and other important papers exploring the auto-association method were written by Bourlard and Kamp [39], and Hinton and Zemel [40]. The first and still one of the main uses today for autoencoders is dimensionality reduction of data, also called lossy compression. Initially, there has been some controversy around the claim that autoencoders could learn more interesting properties of the data compared to PCA. For instance, it was claimed by Bourlard and Kamp [39] that non-linear activations functions in the hidden layer were unnecessary because optimal weights for linear output units could be derived with standard linear algebra. Furthermore, Cottrell and Munro [41], together with Baldi and Hornik [42] claimed that autoencoders produced a compression scheme equivalent to PCA. Instead, Japkowicz [43] showed how non-linear autoencoders with one hidden layer are not equivalent to PCA. More recently, Hilton and Salakhutdinov [44] showed the same for autoencoders with more than one hidden layer. Witten et al. [45] also mentioned that an autoencoder with one hidden layer with a linear activation function will find the same subspace as PCA, assuming a squared-error loss function and normalizing the data using mean centering. This points to the fact that approximation of PCA is just a special case for autoencoders and that they are generally able to learn more flexible and useful transformations.

One of the advantages of autoencoders over other techniques is that an autoencoder is, in fact, a NN. Therefore, autoencoders benefit from the many technical advances made in the last years in the development of deep NNs. A few of the improvements made over the last ten years are reported below. Earlier in this section, it was mentioned how having a hidden layer was a way of explicitly limiting the size of the data representation. Another way to do so is to add a sparsity constraint on the activity of the hidden nodes, imposing that fewer units are active at any given time. Therefore, the autoencoder now minimises the loss function below, where $\Omega(h)$ is a penalty term on the activation of the nodes in the hidden layer.

$$L(x, \hat{x}) + \Omega(h) \tag{2.2}$$

Intuitively $\Omega(h)$ will be high when a lot of nodes are active and low when very few nodes are active. This way a specific node will only activate for a small fraction of the training examples, specializing in a specific kind of input and reacting only to that. A common penalty term is Kullback-Leibler divergence between a Bernoulli random variable with mean ρ, the average activation value desired, and a Bernoulli random variable with mean $\hat{\rho}_j$, the average activation value of a generic hidden node j. This way it is possible to build an autoencoder with a hidden layer bigger than the input layer and still make it learn useful properties of the data. Early work on sparse autoencoders can be found in Poultney et al. [46], and Boureau et al. [47], while more recently a variant called k-sparse autoencoder was presented by Makhzani and Frey [48]. In a k-sparse autoencoder, only the k highest activations are kept and all the others are set to zero. Then, the error is back-propagated only through the k active nodes in the hidden layer. This forces the nodes to narrow down the amount of inputs they react to, becoming more specialized.

A natural next step to the single-layer autoencoder is a multi-layer autoencoder. A multi-layer autoencoder is still a symmetrical network, with an encoder on the left side, a bottleneck middle-layer, and a decoder on the right side. The obvious difference is that the encoder and the decoder have hidden layers of their own. Figure 2.1 shows a generic autoencoder with three hidden layers.

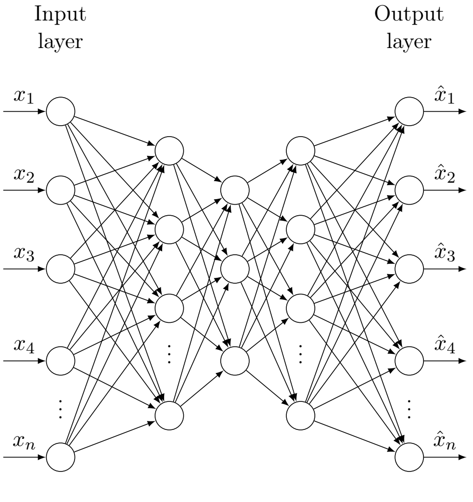

Figure 2.1: Example of a multi-layer autoencoder.

However, the terminology surrounding multi-layer autoencoders is unclear. In literature, it is possible to find references to deep autoencoders and stacked autoencoders interchangeably, but there appears to be a difference. This stance is also corroborated by Hugo Larochelle[1], co-writer of the fundamental paper on denoising autoencoders [49]. Although the terminology is evolving because the field itself is developing at a fast rate, here I stand by

[1] In the video available at the following address: `https://www.youtube.com/watch?v=z5ZYm_wJ37c` the researcher explains the concept of deep autoencoder and specifies in the comments the difference between the two architectures at hand.

the statement that deep autoencoders and stacked autoencoders are different architectures, and their description is as follows. Deep autoencoders are trained as a single-layer autoencoder, while stacked autoencoders are trained following a greedy, layer-wise approach. This second approach trains each layer to reconstruct the output of the previous layer, and then fine-tunes the whole network using the weights found in the pre-training as a starting point [44, 50, 51]. This approach helps with the optimization of the training, by initializing weights in a region near a good local minimum. Experimental results show how multi-layer autoencoders are able to perform a much better compression compared to single-layer autoencoders [44].

Another variant on the standard autoencoder is the denoising autoencoder [49]. While a standard autoencoder minimises a function $L(x, \hat{x})$, a denoising autoencoder minimises a function $L(\tilde{x}, \hat{x})$ where $\tilde{x}$ is the input x corrupted by some noise. Initially, the noise was generated by a Gaussian process, but Bengio et al. [52] empirically proved how that need not be the case. Therefore, denoising autoencoders pursue two different goals: encoding the input to lower dimensional space and recovering data damaged by a stochastic corruption process. The extra goal forces denoising autoencoders to learn more robust features of the input data in order to be able to correctly reconstruct noisier data. Furthermore, regularized autoencoders such as the denoising autoencoder have been shown to implicitly recover the data generating density [53]. Additionally, stacked denoising autoencoders, which are just denoising autoencoders with multiple hidden layers, were used to improve the training of deep NN by learning better starting weights than just random ones [50, 54].

In recent years, the number of NN architectures has expanded significantly, which also means that a number of autoencoders based on such architectures exist. It is not possible to survey all of these architectures, but some references will be given in this paragraph. The main building blocks of autoencoders have already been presented, and the works that will be mentioned will only merge the fundamental concepts with a different, or more specific, architecture. Some of these architectures are: variational autoencoder [55], LSTM autoencoder [56], convolutional autoencoder [57], contractive autoencoder [58].

Chapter 3

Data

This chapter describes the data used to train the autoencoder models. First, Section 3.1 describes the data Crownstone's plugs are able to record. Next, section 3.2 describes the structure of the data contained in available third-party datasets. Then, Section 3.3 describes the preprocessing steps executed to prepare the data for the models. Finally, Section 3.4 presents a description of the anomalies that were used to test the models.

3.1 Crownstone

The main goal of this research project is to understand if it is possible to use autoencoders to discriminate between *normal* and *anomalous* days, using only data coming from appliances connected to a smart plug. The smart plugs produced by the company Crownstone are called Crownstones.

The Crownstones are able to collect several kinds of data that can be used for different tasks, but for the present research only two of these kinds of data are useful. They are listed below.

1. Amount of watts drawn by the monitored appliance at a specific time.

2. "Switch on" and "switch off" events.

Regarding the first kind of data, Crownstones are able to record the Watts drawn by the appliance connected to them at a specific time. Therefore, this kind of data looks like a series of couples composed by a timestamp and an amount of Watts. Regarding the second kind, through the Crownstone application, it is possible to switch on or off the plug, respectively enabling or disabling the appliance attached to it. These events imply an interaction with the user and therefore constitute useful information about their activities.

During my internship at Crownstone, a system was devised to collect data from four houses for a period of three months. Unfortunately, this system was not reliable and the data gathered contained many gaps. Moreover, three months resulted to be too short of a time span for training an autoencoder with the chosen feature, which presented in Section 3.3. Therefore, it was decided to use available third-party datasets compatible with the data that Crownstones collects. The datasets considered are presented in the next section.

3.2 Third-party datasets

In recent years, several datasets were published to allow research in household electric consumption at the appliance-level. Table 3.1 shows the features of some of the datasets [59–62] suitable for the task at hand, where duration denotes the length of the data gathering period, the sampling frequency denotes the amount of time between two measurements, and the number of houses involved in the data gathering process.

Name	Duration	Sampling frequency	No. houses	Extra Information
ECO	up to 8 months	1 minute	6	physical presence
GREEND	up to 1 year	1 second	9	
HES	up to 1 year	2 or 10 minutes	250	
UK-Dale	up to 3.5 years	6 seconds	5	events

Table 3.1: Comparative table of datasets.

The structure of the datasets containing interesting information for the task at hand is displayed in Table 3.2. The displayed information is taken from the UK-Dale dataset, but all the referenced datasets have the same structure. The *Timestamp* column displays information about the time a measurement was taken, represented as a Unix timestamp, and the *Watts* column displays the amount of power drawn at that moment.

Timestamp	Watts
1352744825	0
1352744831	339
1352744837	327

Table 3.2: Structure of the power datasets.

How can this type of data be represented in a graph? Figure 3.1 shows the usage over time of two appliances: washing machine and kettle. The Watt consumption is reported on the y-axis, and time is reported on the x-axis. The blank spaces represent missing data.

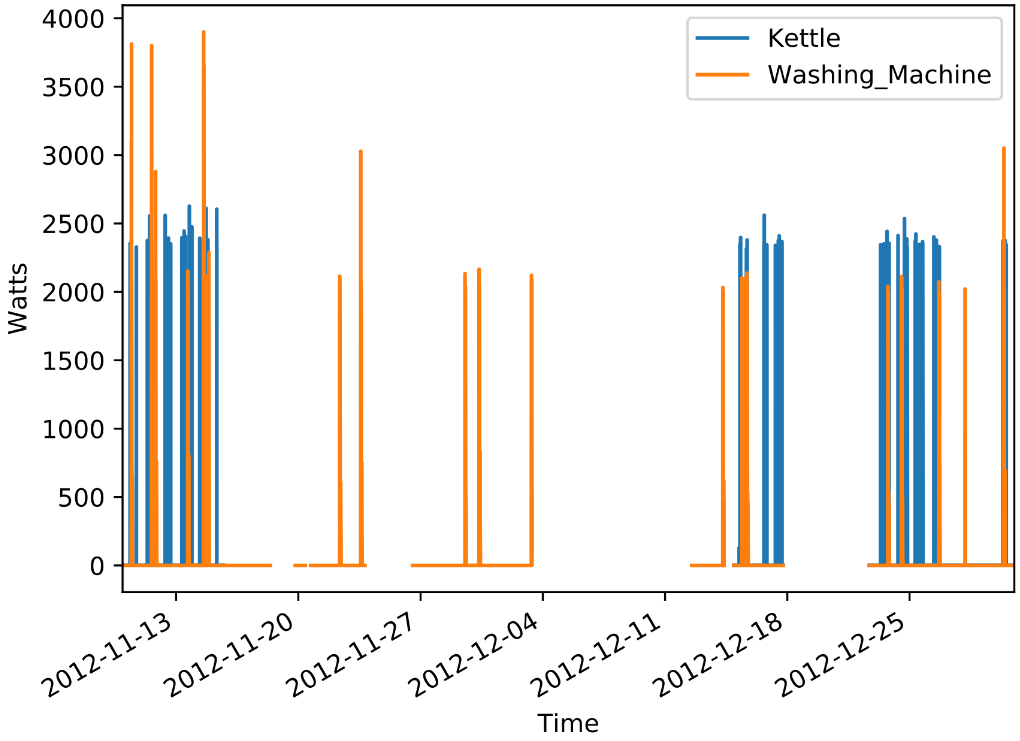

Figure 3.1: Power usage over time of a washing machine and a kettle.

From this simple graph it is already possible to understand a few things about the data at hand:

1. Some data are missing.

2. At this level of granularity, the data look like less structured Poisson distribution, rather than a classic time series like a stock price.

In figure 3.1, the power usage of the appliances looks like a spike. Zooming in and taking a closer look, it is possible to see how each appliance has a different usage of power over time. For instance, a kettle and a washing machine have vastly different power consumptions over time, as it is possible to see in Figure 3.2 and 3.3. The number 30 in the x-axis of both figures represents the day, followed by the hour and minutes in which a measurement was taken.

To the extent of my knowledge, no research has been done on anomaly detection using directly this kind of data. For instance, the approach taken in [18, 19] is to first extract events from raw power data, and then look for

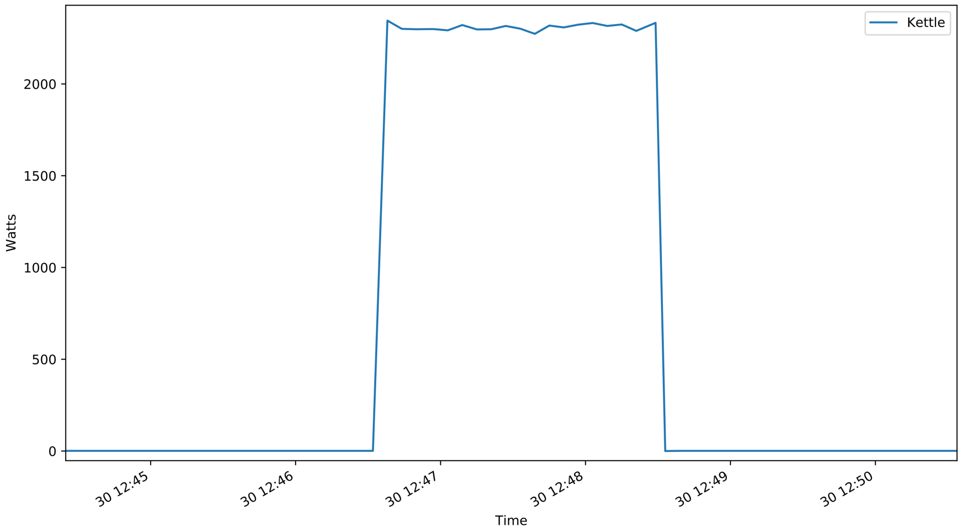

Figure 3.2: Typical power usage over time of a kettle.

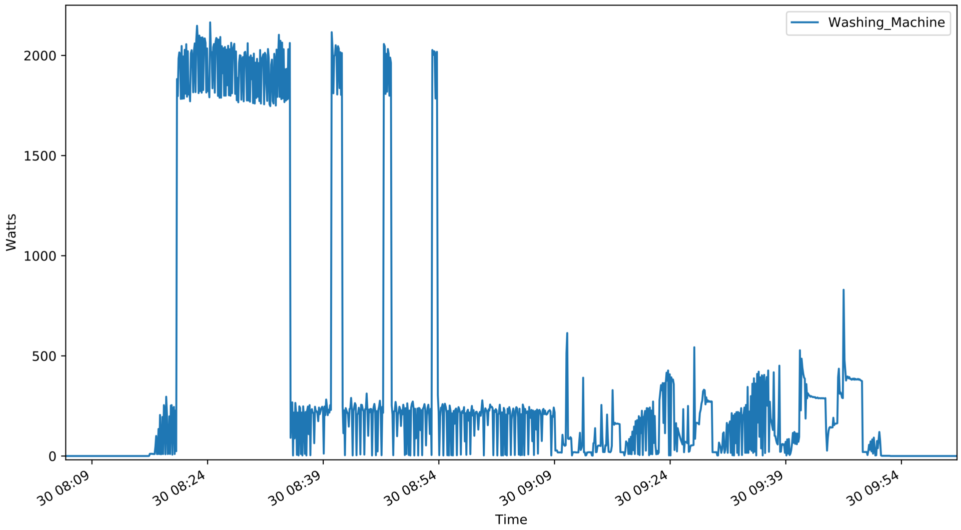

Figure 3.3: Typical power usage over time of a washing machine.

anomalies. The insight that underlies this strategy is that an event would require human interaction; therefore, it reveals something about the activity of the tenant. However, the electricity usage of different appliances can vary wildly, and establishing a way to automatically extract events for every appliance in the dataset is not a trivial matter.

The UK-Dale dataset provides event datasets, in addition to the power datasets of the same appliance, in which “switch on” and “switch off” events of the monitored appliance are recorded. The event datasets have a similar structure to the raw power datasets, as shown in Table 3.3 where 0 stands for “switch off” event and 1 stands for “switch on” event.

Timestamp	**Event**
1364222303	1
1364222428	0
1364223029	1

Table 3.3: Structure of the event datasets.

The authors of the dataset state that “switch on” events should be a perfectly clean recording (i.e. the only possible reason for a “switch on” event appearing in the data is that the user pressed the switch). Unfortunately, “switch off” events may include false positives. But this does not mean that when there is a spike in the power usage there is a “switch on” event, just that when there is a “switch on” event there is also a spike. However, not all appliances are reliably monitored for events: the washing machine dataset and several other appliances miss substantial amounts of data. Figure 3.4 shows the comparison between raw power and event datasets, where a value of 1 in the event graph denotes a “switch on” event, and a value of 0 denotes a “switch off” event.

Because of these complications, and the fact that to the extent of my knowledge no research has been done on anomaly detection using appliance-level power data, it was decided that only the raw power datasets were going to be used. The best dataset for this research is the UK-Dale dataset because it has the longest data gathering process, a high sampling frequency, and a high number of monitored appliances. Therefore, the UK-Dale dataset was chosen for the training, evaluation, and testing of the models.

3.3 Preprocessing

When working with real-world datasets, gaps in the data and inconsistent values are to be expected. To prepare the data for the models, several choices were made regarding how to handle these complications and they will be presented in this section.

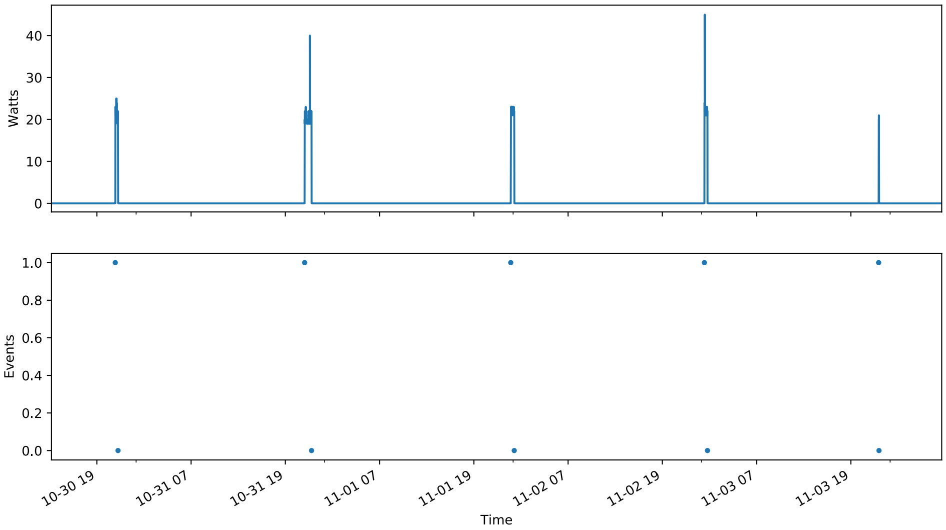

Figure 3.4: Comparison of raw power data and event datasets.

First of all, the choice regarding the level of granularity at which the data should be examined had to be made. The data is spanning more than three years, and at a sampling rate of 6 seconds that is a large amount of data to be considered. Going back to the initial goal of this project, it became clear that the most appropriate unit of time to classify as normal or anomalous would be the day. A longer time unit would make it impossible to act on a potential emergency in a useful time, a smaller time unit would make it difficult to define what an anomaly is. Therefore, the decision was made to select as features the sum of all the values collected for each hour in a day. Aware of the fact that to obtain the average consumption in an hour one should average the power values, the sum was chosen because it gives a more immediate idea of the time in a day when it is more common for an appliance to be used. In any case, the two measures are directly related. Thus, a change to consider the average of the power values instead of the sum can easily be made. Based on these choices, an input vector for the model has the shape $[x_0, x_1, x_2, \ldots, x_{23}]$ where x_0 contains the sum of the power values of all data point measured in a specific date between midnight and one in the morning, and so on for every hour in that day.

As seen in Figure 3.1, data points are missing throughout the whole data gathering period. A threshold was set to 60 minutes of consecutive missing data to determine whether a day should be kept as a valid input or not.

Monitoring an appliance to infer the habits of the tenant relies on the assumption that the monitored appliance is used with some continuity. Therefore, only appliances that are most of the times used at least once a day can be considered useful for this kind of monitoring. Of course, it is possible that a person will not use an appliance for some time. However, more than two consecutive days during which no usage was recorded are considered a vacation. Thus, they are removed from the dataset.

Another extra step to clean the data was to remove the electrical noise picked up by the monitoring device. When an appliance is in standby or even turned off (for the task at hand there is no difference between these two situations because both imply that there was not an interaction with the user) the power value of every data point should be zero. On the contrary, because of electrical noise, some of the data points that were supposed to have value zero had low but positive values. In the UK-Dale dataset metadata, most appliances' descriptions report a power value below which the appliance is considered to be turned off. Every value in each appliance's dataset below that value was then turned to zero. The values above that threshold were not modified because, even if affected by the same electrical noise, on average each data point should be affected in the same way. Therefore, this would not create any problem for the task at hand.

In order to be fed to the model, the input data needed to be split in training, validation and test sets. The respective percentages of the whole dataset are 70%, 15%, and 15%. At the time of splitting, the input vectors are shuffled in order to avoid any temporal interference. If they were not shuffled, the test set would be composed only by the most recent data points. However, if there was a shift in the habits of the user in the period captured by the test set, that could impact the final results and considerations on the model. This is because the training data would be capturing older habits which are not relevant anymore.

3.4 Anomalies definition

The data used in the experiments are described in the previous sections of this chapter. Building on that, this section defines the anomalies that are relevant in this context.

Considering again the wide definition of anomalies given by Chandola et al. [9]: "patterns in data that do not conform to a well-defined notion of normal behaviour", it is possible to highlight two characteristics of this kind of data:

1. Anomalies are context dependent.

2. Anomalous data are defined in contrast to normal data.

The first point indicates that in any different context there might be a different definition for anomaly. The second point suggests that the first step towards the definition of anomaly for a given context is the definition of what data is normal. In the context of the present research, *normal* means frequent, and *anomalous* means rare. To clarify, if the data shows that in the vast majority of days there is a high electricity consumption from a specific appliance between 7 and 8 PM, and a very low consumption between 4 and 5 AM, then a high consumption between 4 and 5 AM on a specific day may be regarded as anomalous for that appliance.

The data coming from a TV is now taken as an example to illustrate the anomalies. Figure 3.5 shows the average TV consumption over all the days in the UK-Dale dataset. It is clear that the data shown have the approximate shape of a Gaussian distribution with mean around 20. Again, anomalies are dependent on the context and for this research four kinds of anomalies have been devised, trying to cover most of the possible cases. More kinds of anomalies can, of course, be devised but as a starting point the four reported below should suffice. The anomalies were generated by:

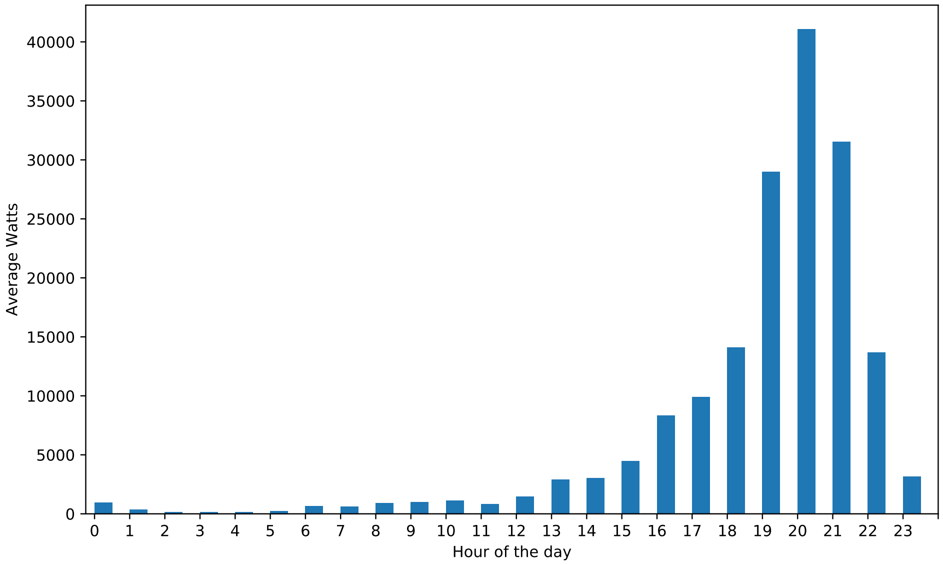

Figure 3.5: Average daily distribution of TV's power consumption.

1. Shifting the time of the power consumption by a number of hours x, such that $1 \leq x \leq 23$.

2. Scaling all data points by different factors.

3. Combining point 1 and 2.

4. Flattening consumption to specified values.

The anomalies defined in point one cover all cases in which the routine of the resident shifts in time. Some of these shifts could be dangerous, such as an increased power consumption in late evenings and night. This behaviour could be a symptom of *sundowning*, often associated with Alzheimer's disease and dementia [63]. An anomaly of this kind is presented in figure 3.6, where the same values presented in figure 3.5 are shifted by 8 positions to the right. The anomalies defined in point two consider the cases in which the habits of the tenant remain the same from the time perspective, but vary in intensity. The scaling factor chosen for this type of anomaly are 0.5, 0.75, 1.25, 1.5, 2. The meaning of these anomalies depends on the appliance being monitored: for instance, in the case of a cooking stove, if there is half the usual amount of power drawn for several consecutive days, it may mean that the eating habits are changing. The anomalies defined in point three combine a shift in time to a scaling factor. The anomalies defined in point four describe situations in which an appliance is left active and continuously draws a certain amount of power. In the case of a TV being always on, it may mean that the tenant is having a more secluded and sedentary lifestyle. Specifically, the values chosen for each day are the minimum, the maximum, and the mean for that day.

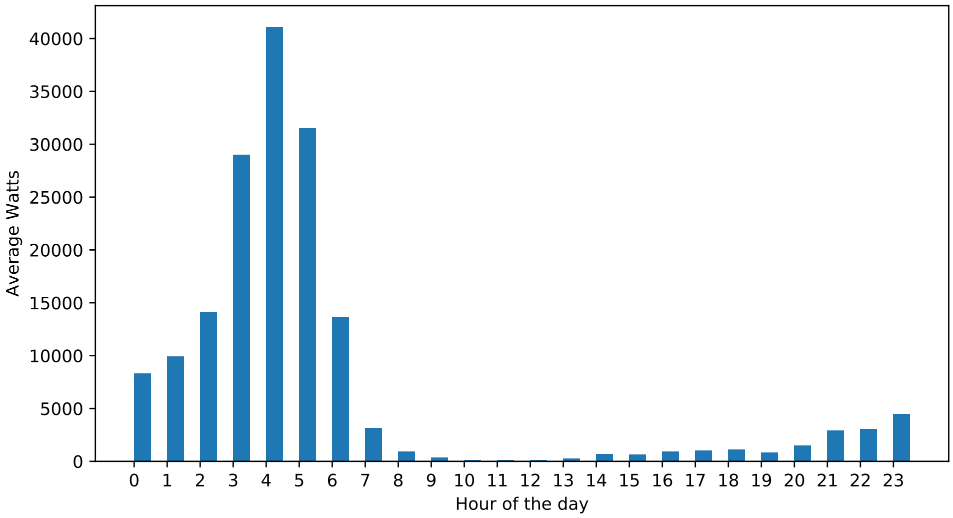

Figure 3.6: Average daily distribution of TV's power consumption shifted by 8 positions.

Chapter 4

Method

This chapter describes how the final model is chosen, trained, and evaluated. First, Section 4.1 illustrates the steps taken in order to answer every research sub-question initially presented in Section 1.3. Then, Section 4.2 discusses the network architecture and its hyperparameters. Finally, Section 4.3 presents the choices made for the evaluation of the models.

4.1 Experimental steps

In order to answer each research sub-question specified in Section 1.3, several experimental steps were taken and are reported below.

To answer sub-question 1, *"What kinds of anomaly are the models able to recognize?"*, a trained autoencoder is tested on the anomalies generated according to Section 3.4. The specific autoencoder selected for the task was the result of the answers to sub-questions 1 and 2, and is described in Section 4.2.

To answer sub-question 2, *"How does performance evolve when the size of the latent representation is increasingly reduced?"*, several models with increasingly smaller latent representations are trained and tested with the anomalies devised in Section 3.4. Results are then analysed to understand the performance changes.

To answer sub-question 3, *"Is there a clear performance improvement when a deep autoencoder is used instead of a shallow autoencoder?"*, a seven-layer autoencoder is built and tested against its shallow counterpart to investigate the different performance on the anomalies.

4.2 The network and its hyperparameters

In the context of machine learning, the term hyperparameter refers to those parameters that are set before, and not influenced by, the process of learning. In the case of a NN for instance, the hyperparameters define the architecture, the number of nodes per layer, batch size, learning rate and many other aspects of the model. Unfortunately, there is no reliable method to select the best values for the hyperparameters. The most common approach is training a model with different values and testing it, to find the set of values that best suit the uniqueness of the dataset and the desired result [45].

Hyperparameter tuning is one of the tasks mainly responsible for the success of a machine learning model. It is a process that can take a long time and a lot of resources, so much so that state-of-the-art models are able to perform so well because of the huge amount of computational power that is available to the researchers and/or the company supporting them. Because of the limited resources available for this research, only a few of these hyperparameters have been explored, and are reported in Table 4.1. The main hardware and software components used to run the experiments are illustrated in Appendix A.

Hyperparameter	**Shallow autoencoder**	**Deep autoencoder**
Layer size	[24, 12] [24, 8] [24, 4]	[24, 20, 16, 12] [24, 18, 12, 8] [24, 17, 10, 4]
Layer activation	[relu, relu] [relu, linear]	[relu, relu, relu, . . . , relu] [relu, relu, relu, . . . , linear]
Feature scaler	Standard scaler MinMax scaler None	
Loss	Mean Squared Error Mean Absolute Error Root Mean Squared Error	

Table 4.1: Model hyperparameters.

The autoencoder is symmetrical, thus the layer sizes are reported for half of the autoencoder imagining it going from left to right as in Figure 2.1. The size of the bottleneck layer is initially set to half of the size of the original input, which is generally considered a good place to start for dimensionality reduction. Then, the size of the bottleneck layer is increasingly shrunk to see the evolution of performance on the anomalies. In the deep autoencoder, the size of the bottleneck layer is the same as in the shallow autoencoder, with the layers decreasing in size by approximately the same amount in each layer. Regarding the activation functions, REctified Linear Units (RELUs) [64] are extremely popular in deep learning and also very appropriate for the data at hand, which have always positive values. The linear activation in the output layer is included because deep networks can suffer from the dying RELU problem. The dying RELU problem is well explained by Andrej Karpathy in the Stanford CS231n course notes [65]. A linear activation in the output layer can help the gradient flow better and avoid that problem. In the current research features are always in similar scales. This is because they are sums of positive values representing hourly usage of an appliance. Therefore, scaling could theoretically be avoided. On the other hand, scaling can help the neural network converge faster and find better local optima. Thus, two scalers are selected for the hyperparameter search: the Standard scaler standardize features by removing the mean and scaling to unit variance, while the MinMax scaler transforms the data by scaling each feature to the range $[0, 1]$.

The loss functions define how the error between the output of the network and the desired value should be calculated; this is the function that is being minimized during training. In general Mean Absolute Error (MAE) is the easiest to comprehend because it expresses values in the same unit of measure of the data. Therefore, it was chosen as the metric to compare the performance of different models. Interestingly, minimizing the MSE almost always generates lower MAE values compared to when the network is minimizing MAE directly. For reference, formulae for MAE and MSE are reported in Equation 4.1 and 2.1, respectively. It is clear that in the case of an outlier, its MSE value will be much larger than its MAE value. This is because an outlier is characterized by a large reconstruction error, and squaring it would make it even larger. Therefore, MAE is considered a metric that is robust to outliers, while MSE is considered a metric sensitive to outliers. Because of this sensitivity, the MSE loss function will adjust the models to incorporate the outlier values, possibly at the cost of degrading performance for data points with smaller reconstruction errors. During backpropagation with MSE loss, minimizing the reconstruction error of outliers is a fast way to decrease the overall MSE value. This is because the value of the metric can be decreased significantly by modifying the reconstruction of a small number of data points. Depending on the number of outliers and their values, it is possible that incorporating them in the model to some degree will also affect the MAE value that is used to evaluate the model after training, thus making it smaller than when minimizing the reconstruction error directly with MAE loss.

A choice that has not been tested through the hyperparameter optimization is the one for the optimizer. For an overview on the available optimizers see [66]. The optimizer Adam [67] was selected because it is generally regarded as the best initial choice [66], also pointed out by Andrej Karpathy in the Stanford CS231n course notes [65]. For this research, the parameters of the optimizer are kept to the default values as suggested in the original paper [67].

For training the deep autoencoder, two methods are investigated: standard whole-network training, and greedy layer-wise training. Greedy layer-wise training [50] consists first in training each layer to the left side of the bottleneck layer as if it was the hidden layer of a single-layer autoencoder, saving its weights and biases. Once the bottleneck layer is reached, all weights and biases for all layers have already been calculated because the network is symmetrical. These weights and biases will be the starting weights and biases for the whole-network fine tuning. This method is more computationally expensive but was able to obtain reconstruction errors that were on average 15% smaller compared to the standard method. Therefore, the greedy layer-wise method was chosen to train the deep autoencoders in the experiments.

In order to define which values for the hyperparameters to use in the experiments, a grid search is performed on the hyperparameters previously mentioned. Then, the best performing model for each bottleneck layer size is selected. The selection of the best model was based on the reconstruction error obtained on the validation set. The validation set is also used to stop the training of the network. Specifically, the training is stopped when the network produces no improvement on the reconstruction error of the validation set for 10 consecutive epochs. Interestingly, a single hyperparameters' configuration performed best for most of the cases. Namely, no scaling, only RELU activations, and MSE loss. Only the deep autoencoders with bottleneck layer of size 8 and 4 were able to best perform with a MAE loss. It needs to be pointed out that most of the runs that used Root MSE as a loss function were inconclusive because they suffered from the exploding gradient problem [37]. The grid search was performed on the TV dataset, the average day of which is represented in Figure 3.5.

4.3 Evaluation choices

Contrary to supervised learning, the performance of an unsupervised or semi–supervised learning model is not as straightforwardly evaluated. While supervised learning already has an answer and therefore one can compute metrics like precision and recall, with unsupervised learning one can only explore the results of the model and try to understand which cases are correctly recognized as anomalies and which cases are missed. This research uses the anomalies as defined in Section 3.4.

As previously done by Martinelli et al. [27], the present research uses thresholds on two different metrics to determine whether a day is anomalous. First, the MAE over all the features is reported in Equation 4.1, where y is one feature vector, $\hat{y}$ is the feature vector reconstructed by the model, and i refers to one of its features. MAE is chosen as a metric, instead of MSE for instance, because the error is expressed in the same unit of measure as the data, and therefore more intuitively evaluated. The MAE is useful to identify as anomalies data points in which the majority of features are reconstructed poorly.

$$\text{MAE}_y = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{4.1}$$

Second, the Mean Absolute Deviation of the Errors (MADE) of all the features from their mean is reported in Equation 4.2, where y is one feature vector, $\hat{y}$ is the feature vector reconstructed by the model, and i refers to one of its features. Certain features might be more sensitive to anomalies than others. Therefore, the MADE metric is useful to identify as anomalies data points that have MAE values below the anomaly threshold, but in which a small subset of features generated most of the reconstruction error.

$$\text{MADE}_y = \frac{1}{n}\sum_{i=1}^{n} \Big| |y_i - \hat{y}_i| - \text{MAE}_y \Big| \tag{4.2}$$

The two metrics identify different kinds of reconstruction error. Therefore, if any of these two values exceeds its threshold, the day is considered an anomaly.

Once the metrics are chosen, a threshold value for each of them must be defined. If a threshold is overcome, then that data point is considered an anomaly. As discussed in Section 6.1, most of the literature investigating the use of autoencoders for anomaly detection is not clear about how the thresholds should be chosen. Dau et al. [68], choose as a threshold the maximum reconstruction error obtained on the training set. This seems like a sensible choice if the training dataset does not contain any anomaly or outlier. Unfortunately, with the chosen dataset there is no absolute certainty that the training data completely belongs to the normal class. It will belong to the normal class in the vast majority of the cases but not necessarily in all of them. Therefore, when setting the threshold on the metrics this must be taken into consideration. This is why the anomaly threshold selected is the 95th percentile of the MAE values of the training set, instead of the maximum value. The choice of that value is arbitrary but motivated by data exploration. Figure 4.1 presents a histogram of the frequencies of the MAE values of the TV's training set and its anomaly threshold. The values were obtained with the shallow autoencoder with a bottleneck layer size of twelve.

As it can be seen, the big majority of the values are low (i.e. below 500), which makes sense given that most of the data points have low reconstruction errors. Because of this imbalance in the distribution of the data, choosing a threshold based on the standard deviation from the mean is not a good option. Figure 4.2 shows the sorted MAE values and the threshold anomaly for the training set of the TV dataset, obtained with the shallow autoencoder with a bottleneck layer size of twelve. As illustrated, the 95th percentile chosen as anomaly threshold is placed at the beginning of the steep

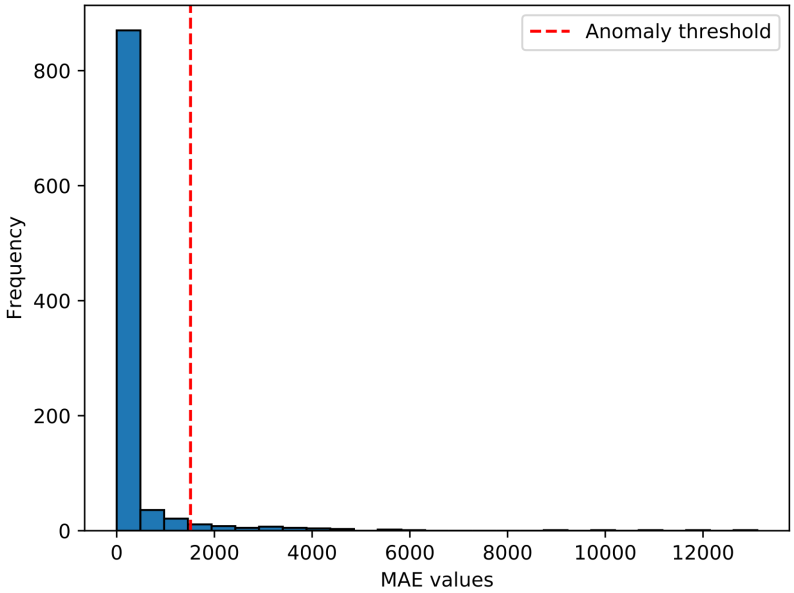

Figure 4.1: Histogram illustrating the frequencies of the MAE values of the TV's training set.

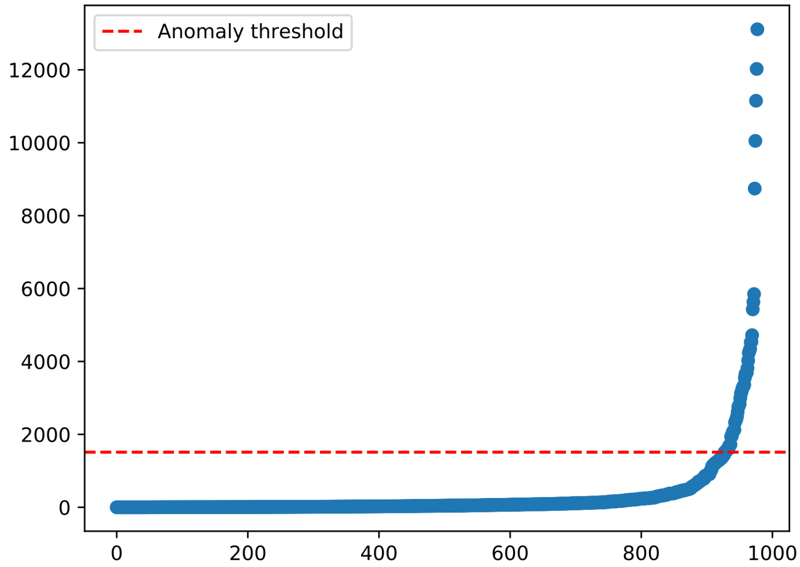

Figure 4.2: Sorted MAE values of the TV's dataset.

exponential curve. This allows for some higher values to be still considered normal, but not the extremes. Because the data distribution is very similar, the same choice is made for the MADEs; the 95th percentile is chosen as the anomaly threshold.

As a concluding remark on the methodology, it is important to clarify that the final system is composed of two main parts: the trained network and the threshold. The present research focused more on the network as a component but, as highlighted in Section 5 and 6, a good threshold is paramount to the anomaly detection capabilities of the system. More research needs to be done in order to obtain some guidelines to select an appropriate threshold that is not only based on the exploration of the dataset at hand.

Chapter 5

Results

This chapter presents the results obtained by the models specified in Chapter 4. First, Section 5.1 presents the results obtained on the TV dataset. Then, Section 5.2 the results obtained on the microwave dataset. Finally, in Section 5.3 some considerations on the performance of the models are presented.

There are several reasons why the datasets of TV and microwave have been chosen. Upon inspection, the average daily consumption of an appliance generally has a shape which approximates a Gaussian distribution or a mixture of Gaussian distributions. Therefore, TV and microwave have been selected as examples of these two average shapes. Moreover, they are appliances with a low rate of interruptions in the data, which means more data to train the models. Finally, the rate of empty days after the preprocessing was 4.2% for the TV and 5.1% for the microwave. This means that the appliances were used almost every day, which is ideal for the scenario this research is focusing on.

In order to avoid redundancy while reporting the results, each model will be given a code name based on its architecture–shallow or deep–and the size of the bottleneck layer: 4, 8, or 12. To clarify, the shallow model with bottleneck layer size of 8 will be referred to as shallow8. Moreover, all the results and figures will be referring to the TV dataset, unless explicitly stated otherwise.

5.1 TV dataset

This section is dedicated to the analysis of the performance of several models on the TV dataset. To reiterate, performance in this context is not a straightforward concept. The comparison between models is made by exploring the data that each model deemed anomalous, and by considering their differences. Moreover, because NN models appear as a black box, it is impossible to know for each instance the reason the model performed in the way it did. On a general level it is trying to minimize the reconstruction error, but for specific examples we can only suppose that the patterns that caused the high reconstruction error were not adequately represented in the training data. The average usage of this appliance is reported in Figure 3.5. This specific appliance was chosen because of its average usage shape–almost a Gaussian distribution–but it does not mean that the all of the input vectors actually have a resembling shape.

5.1.1 Test set results

This section presents a performance comparison between shallow and deep models with shrinking bottleneck layers trained on the TV dataset. Before going into the specifics of each comparison, there are a few general considerations worth pointing out. First, no model is able to properly reconstruct consumption that happens between midnight and six in the morning. As illustrated in Figure 3.5, few data points have high values in that hour range, which explains the poor reconstruction by all models. Second, the smaller the size of the bottleneck layer, the smaller the size of the latent representation. Therefore, the poorer the reconstruction on examples that are rarely seen in the training data. Because of how the anomaly thresholds are computed, this also leads to a higher threshold for anomalies in models with smaller latent representations, as it can be seen in Figure 5.1.

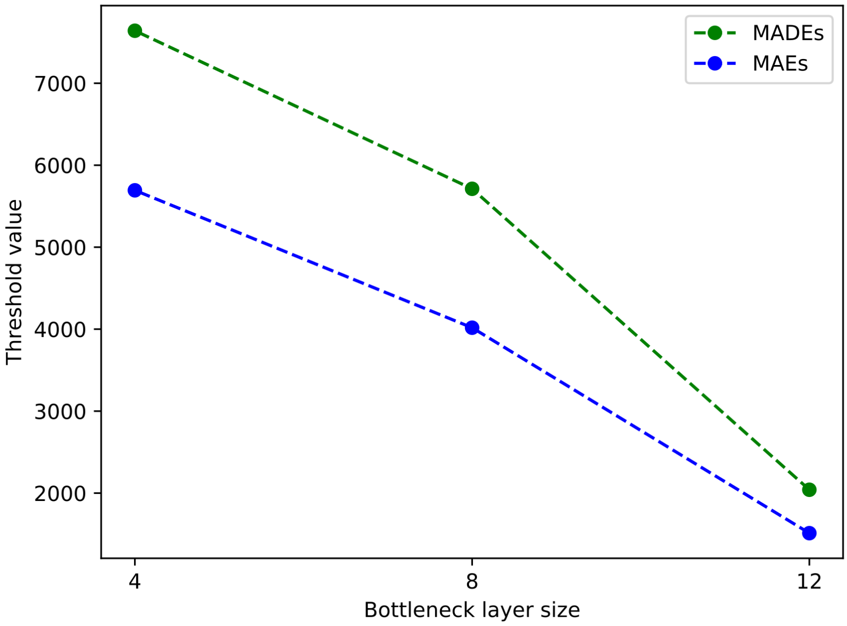

Figure 5.1: Anomaly threshold values per bottleneck layer sizes.

Moving on to the comparison of the shallow models with shrinking bottleneck layers, the first comparison will be between shallow8 and shallow12, starting with the days recognized as anomalies in the test set. First, consider the anomalies detected by shallow12 but *not* by shallow8. One reason that causes some days to be anomalous for shallow12 but not for shallow8, is that anomaly thresholds are higher in shallow8, as seen in Figure 5.1. So even if the reconstruction of the two models was similar, the stricter thresholds in shallow12 triggered the anomaly. An example of this behaviour is reported in Figure 5.2. For this example, the behaviour of shallow12 is correct because both reconstructions completely missed the early morning consumption, shown in Figure 5.6. Thus, the day should be considered anomalous. In few occasions, days were not detected by shallow8 because this model was better than shallow12 at reconstructing high values around midnight. This behaviour was unexpected, and not in line with Hypothesis 2. An example of this behaviour is reported in Figure 5.3. The value that the models were trying to reconstruct in the first hour of the day was 31415, and shallow8 does a better job. On the other hand, shallow8 is consistently unable to reconstruct well values in the hours ranging from 13 to 16. Therefore, it triggers more alarms compared to shallow12 at that time of the day. An example of this behaviour is shown in Figure 5.4. With shallow4 the situation gets even more extreme. It reconstructs badly inputs that shallow12 is able to reconstruct very well. As with shallow8, the time range 13–16 is very poorly reconstructed, and on top of this also the time range 17–19 has big troubles. In general, it fails to properly reconstruct days that have a high consumption distant from the evening hours. A fairly unusual day like the one reported in Figure 5.5 is very badly reconstructed, highlighting the fact that only the values around 20–21 are reconstructed relatively well.

Now the deep autoencoder models will be considered. Generally speaking, there is nothing to add to the behaviour of the shallow models. The differences between deep12 and deep8 are the same as the differences between shallow12 and shallow8. Namely, high consumption in the time range 13–16 triggers deep8 but not deep12. The parallel continues with deep4: in addition to shallow8 limitations, also peaks of consumption in the time range 16–18 are very poorly reconstructed. As for shallow4, when deep4 is not triggered by days that triggered models with bigger latent representations it is because its thresholds are higher.

From this analysis of the results, it appears that the models with latent representations of size four and eight nodes are too rigid. On the other hand, the models with latent representation of twelve nodes are more balanced. Therefore, they are chosen for the comparison between shallow and deep models. Deep12 is able to reconstruct better than shallow12 consumption in the morning. While shallow12 is not able to reconstruct it almost at all, deep12 is able to reconstruct it to a certain degree, as shown in Figure 5.6. In this case, even an isolated peak in the morning is reconstructed well enough

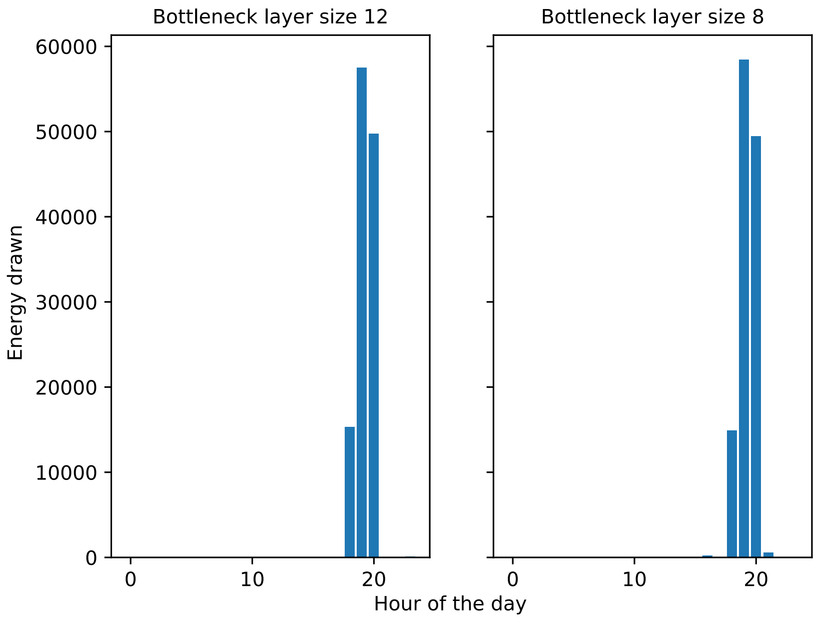

Figure 5.2: An example of similar reconstructions performed by shallow models with bottleneck layer size of twelve and eight nodes, respectively.

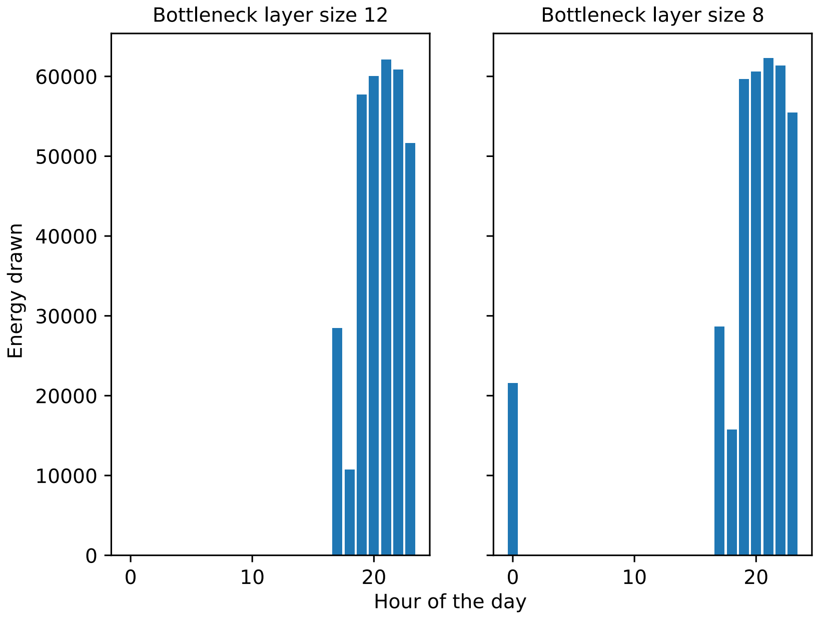

Figure 5.3: An example of different reconstructions around midnight performed by shallow models with bottleneck layer size of twelve and eight nodes, respectively.

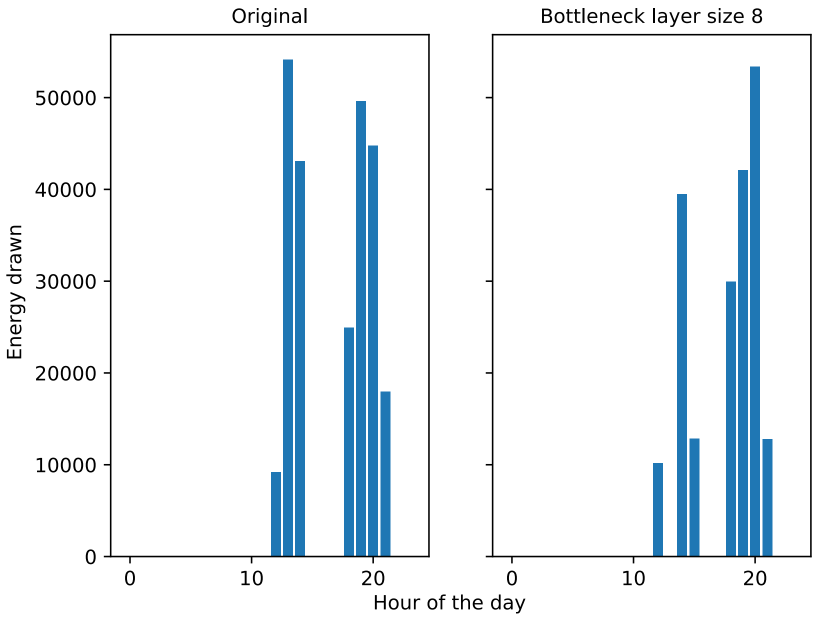

Figure 5.4: An example of poor reconstruction of the 13–16 time range performed by the shallow model with bottleneck layer size of eight.

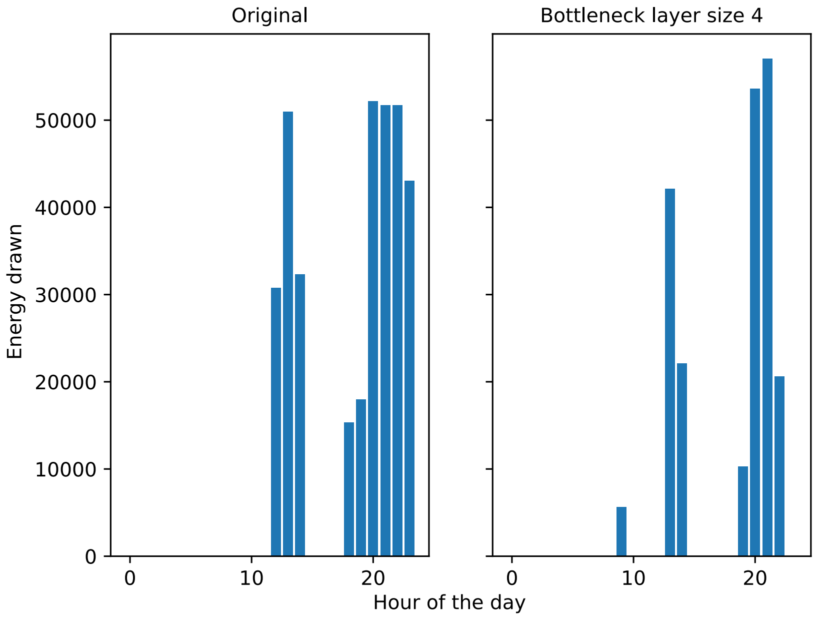

Figure 5.5: An example of the fact that the shallow model with bottleneck layer size of four is exclusively able to reconstruct the time range 20–21.

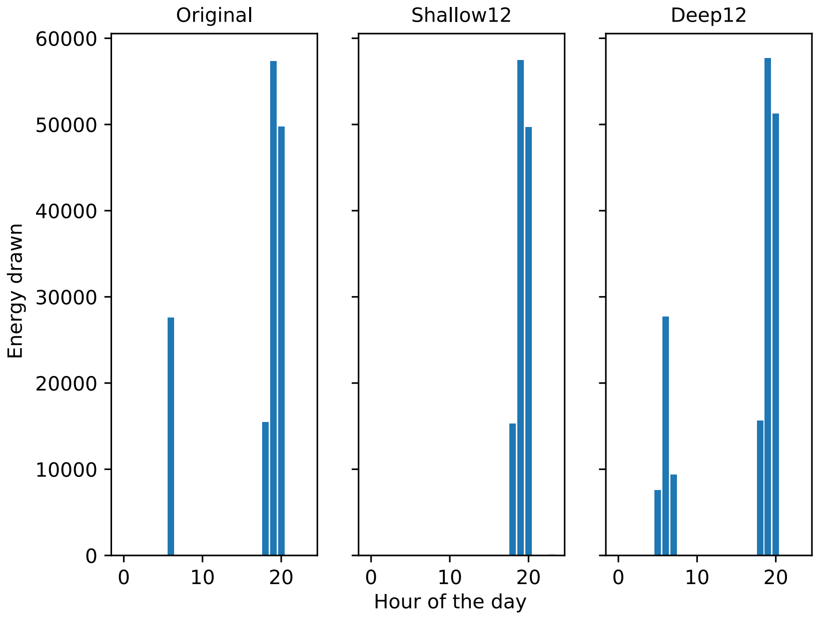

Figure 5.6: An example of early morning reconstruction poorly performed by shallow12. Deep12 performed well enough not to overcome the anomaly thresholds.

by deep12 not to overcome the anomaly thresholds. On the other hand, in a few cases vanilla12 was able to better reconstruct peaks in the range 11–15, as shown in Figure 5.7. This behaviour was unexpected, and not in line with hypothesis 3. The expectation was that deeper models would learn a more flexible representation of the data, and therefore would be able to reconstruct the input better than shallow models. However, in this case the behaviour of shallow12 is more appropriate because it is much more likely to see TV consumption around lunchtime than it is to see it at six in the morning, at least in this dataset. Considering the above result, it appears that overall shallow12 is able to find a better degree of flexibility compared to deep12.

5.1.2 Anomalies results

This subsection focuses on the analysis of the models' results on the anomalies artificially generated according to Section 3.4.

The anomalies in which the time of the day is shifted by increasing values, as illustrated in Figure 3.6, will be referred to as the *rolled* anomalies. First, a comparison of the models' shallow12 and deep12 performance on the rolled anomalies is provided. Figure 5.8 shows a box plot of the average MAEs values for each of the 23 shifts in this category of anomalies. Position

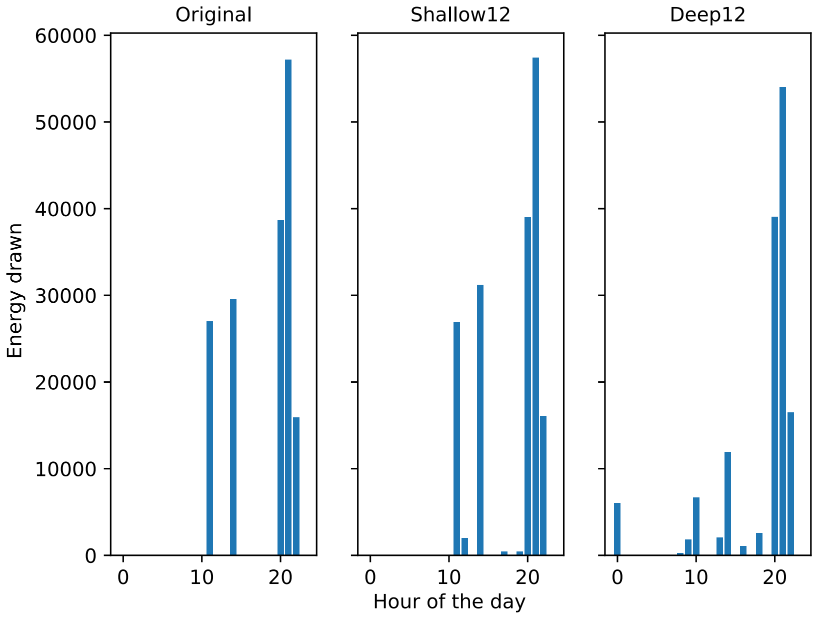

Figure 5.7: An example of the reconstruction of the time range 11–15 performed well by shallow12 but not by deep12.

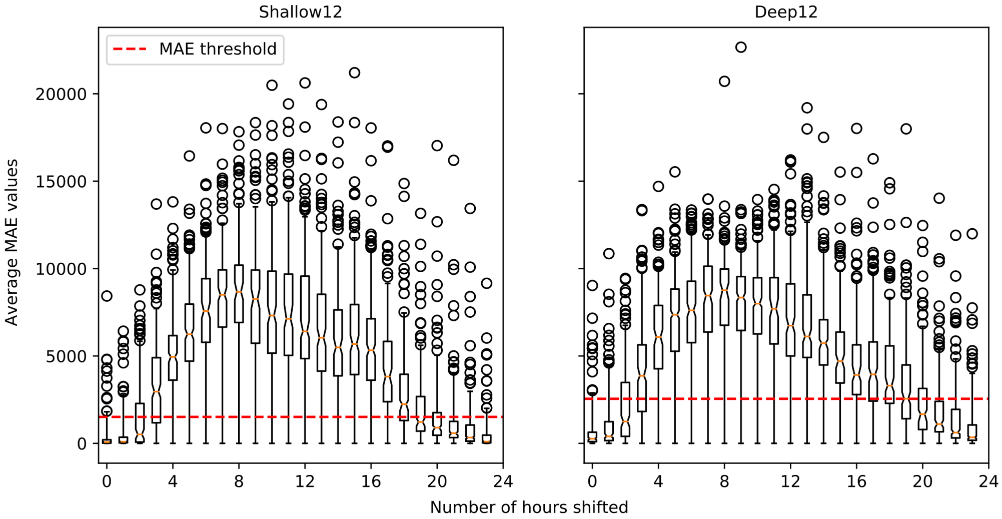

Figure 5.8: Box plots of the average reconstruction error per shift in the rolled datasets, obtained by deep and shallow models with bottleneck layer size of twelve.

0 reports MAE values for the original test set. The notches around the median represent the 95% confidence intervals obtained by bootstrapping the median 5000 times. As usual, the box contains the values ranging from the first to the third quartile. The lower whisker goes from the value zero to the first quartile, and the higher whisker goes from the third quartile to the 95th percentile. As it is possible to see from Figure 5.8, the anomaly detection threshold coincides with the end of the upper whiskers in the original test set, which is in position 0. There is not much difference in these two graphs between the shallow and deep models, except that the 95th percentile seems to be closer to the third quartile in the range 7–11 for the deep model. Another small difference is that the boxes are relatively more distant from the anomaly threshold in the shallow model. No clear distinction in the performance between the models shallow12 and deep12 was found. Therefore, from this point forward only the shallow model will be considered, given that is easier and faster to train. Intuitively, the shape of the box plots makes sense because the average error values increase with the size of the shift until it reaches a maximum. In the case of the shallow model it is a shift of seven positions. Then, because the data distribution is similar to a Gaussian, the average errors decrease because the peak of the Gaussian is approaching again its original position. For the TV dataset, it is clear that a positive shift of two hours in the user's habits already causes a steep increase in the average errors, and results start to get similar to the original test set only after 22 shifts. Therefore, this means that a shift of two hours in any direction already increases a lot the probability that a day is considered anomalous. Specifically, anomalous days in the normal test set constitutes 7.6% of the total, while anomalous days in the test set shifted of two hours constitutes the 36.2%. Figure 5.9, shows the number of anomalies detected for each shift. It is interesting to see how the number of anomalies starts increasing sharply after the first one-hour shift. Then, it flattens out around 200 from a shift of four hours to a shift of sixteen hours. Finally, it decreases almost as sharply as it increased in the beginning. This happens because, out of 210 days in each shift, 7 are perfectly empty and a few others have very low consumptions. This means that their reconstruction errors are very low and therefore they are not considered as anomalies. In a few cases, anomalies are missed for another reason. As it is possible to see in Figure 5.10, the reconstruction is suboptimal. However, at the same time, the relatively low consumption allows the reconstruction error of the peak to be spread over all the other features and not be considered anomalous. For cases such as this, it could be useful to introduce alarms for very high deviations in single features instead of averaging over all features, as per the Formula 4.2.

Now the scaled anomalies will be considered. As explained in Section 3.4, in this kind of anomaly every vector in the test has been scaled by several different factors. Then, each vector is tested to see if it is identified as an anomaly. Analysing the results, it turns out that an almost exact linear

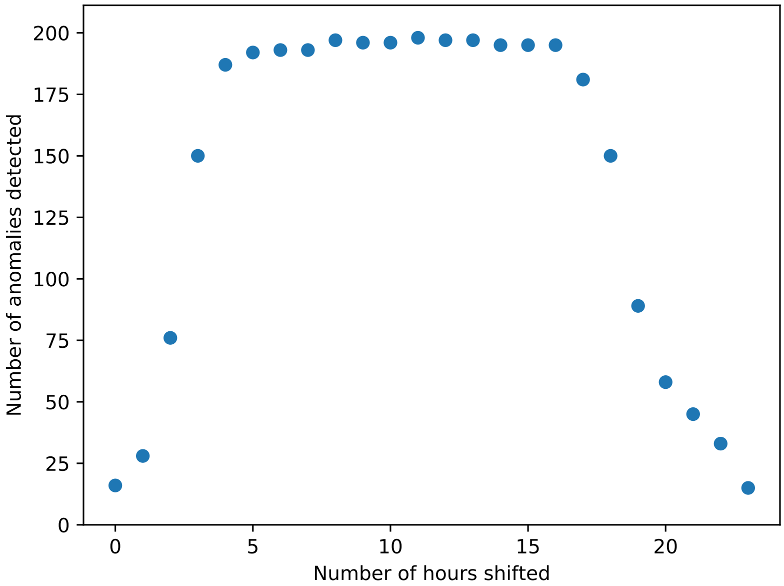

Figure 5.9: Number of anomalies detected per shift in the rolled datasets by the shallow model with bottleneck layer size of twelve.

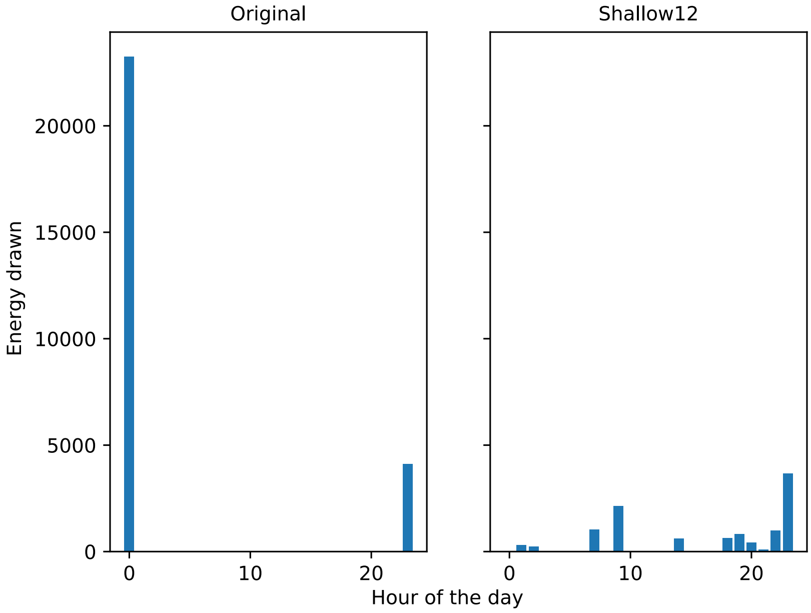

Figure 5.10: An example of a poorly reconstructed day which is not considered an anomaly.

relationship exists between the errors of the test set and the errors of the scaled test sets. This is evident in Figure 5.11, which shows the relationship between the sorted MAE values of the test set and the respective MAE values of the test set scaled by a 0.5 factor. This relationship holds for MADEs as well, and for every scaling factor. Intuitively, this makes sense because the lower the values to reconstruct, the lower are also the errors. This is the case when trying to reconstruct days that are empty or that have a very low consumption. But this relationship makes it true for high values as well, so if one scales an input by a factor 2, its errors value will be two times larger than the original. It follows that the lower the scaling factor, the more points below the anomaly threshold there will be. Vice versa, the higher the scaling factor, the more points above the anomaly threshold there will be. The relationship between scaling factor and the number of data points recognised as anomalies is not clear. As it can be seen in Table 5.1, the scaling factors smaller than 1 maintain approximately a linear relationship, while the scaling factors bigger than 1 do not.

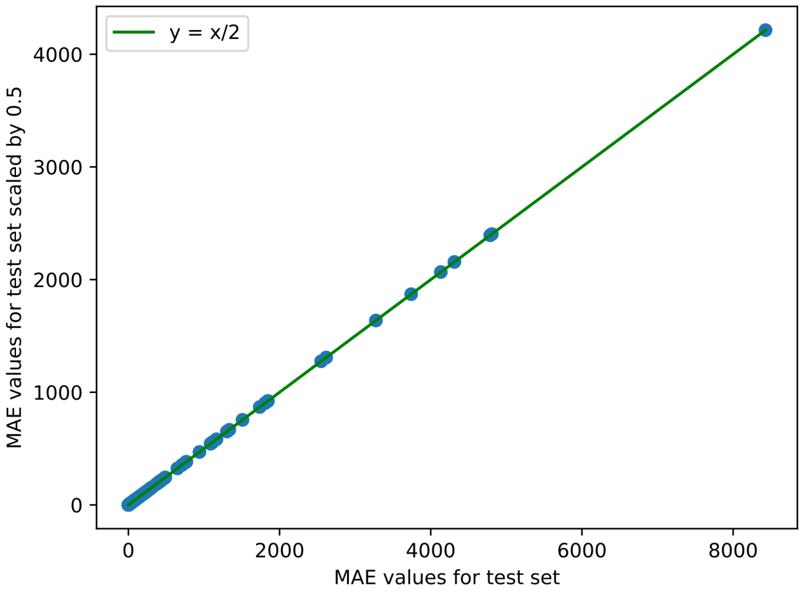

Figure 5.11: Relationship between the MAEs of the test set and the MAEs of the test set scaled by a 0.5 factor.

The combination of the previous two types of anomalies did not bring forth anything additional to the single analyses. The linear relationship between the MAEs of the test set and the MAEs of the scaled sets is evident also in Figure 5.12 because the average error values of the scaled test sets are the original error values multiplied by the respective scaling factor. In

Scaling factor	Number of anomalies
0.5	9
0.75	11
1	16
1.25	17
1.5	20
2	23

Table 5.1: Number of anomalies detected in the scaled test sets.

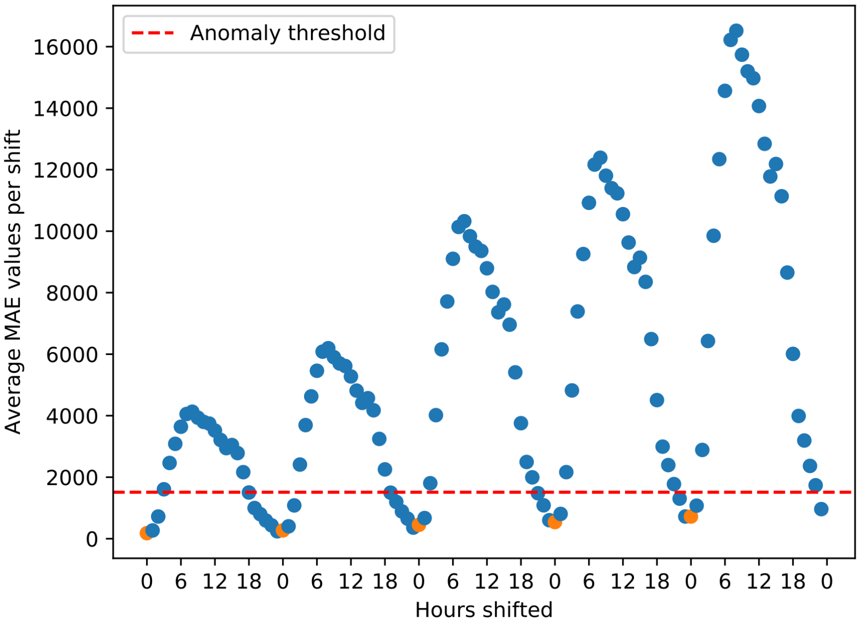

Figure 5.12: Average MAE values per shift in the rolled anomalies and per different scaling factors.

this figure every curve is the simplified version of Figure 5.8, in which every point is the average MAE for that shift. The scales are the one reported in Table 5.1, and the orange points represent position 0 of the plot of a scaling factor. As for the simply scaled anomalies, the lower the scaling factor, the more points below the anomaly threshold there are.

Finally, the last type of anomaly is considered. This type of anomaly consists of selecting a single value from a specific day and substituting all other values with that one. This way, every value in a day is the same. For each day, three values are selected: the minimum, the maximum, and the mean of that day. The analysis of this type of anomalies does not add anything significant to the information gathered from the previous analyses. The days flattened at the maximum value are always recognised as anomalous, as long as the original values are not all zero or extremely low. This makes sense because the average shape is similar to a Gaussian, while this type of anomalies has high values at every hour. Similarly to the days flattened at the maximum value, the days flattened at the mean values are always recognized as anomalous, except days in which the value is low or zero. Every day containing values smaller than 2000 per hour is considered not anomalous. On the other hand, the days flattened at the minimum value are never recognized as anomalies because the minimum values are almost always zero or very low. Therefore, the network produces very little reconstruction error and the anomaly thresholds are not overcome.

5.2 Microwave dataset

The shape of the microwave's dataset average daily power consumption is different compared to the TV's dataset. The microwave dataset approximates a mixture of three Gaussians, instead of a single Gaussian, as illustrated in Figure 5.13. This will affect the performance of the models, therefore the results of shallow12 are reported here. During the analysis of the TV dataset results, shallow12 appeared as the more balanced model, therefore it was chosen for this analysis as well.

Shallow12 detects 13 anomalies in the test set, corresponding to 6.3%. Upon inspection, two recurring characteristics of these anomalies appear clear and are reported as follows. The model is not able to reconstruct well peaks in the night and in the early morning, as shown in 5.14. Moreover, consumption in the range 21–22 is poorly reconstructed, as illustrated in Figure 5.15.

Regarding the scaled sets, there is nothing more to add to what has been said about the TV dataset. Moreover, the model maintains a linear relationship between input scale and error reconstruction values.

Regarding the rolled anomalies, Figure 5.16 shows the box plot picturing the average MAE values per shift. The parameters of these box plots are

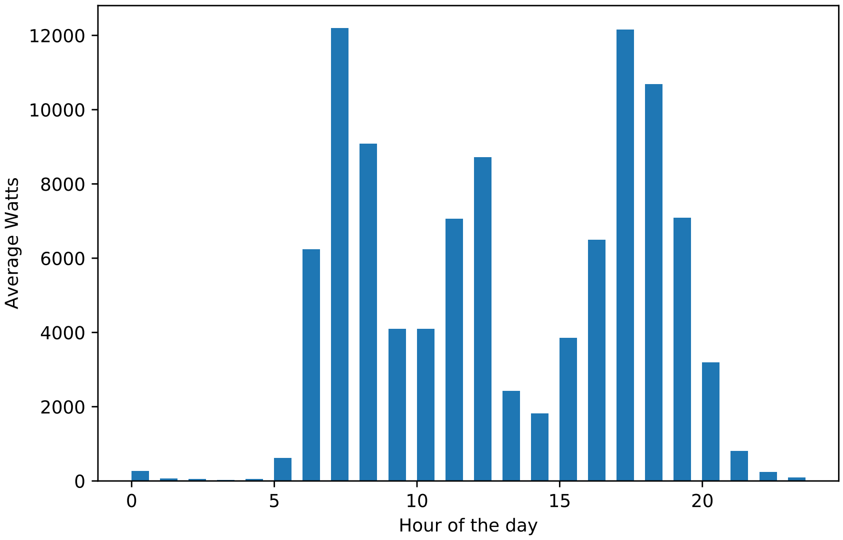

Figure 5.13: Average daily distribution of microwave's power consumption.

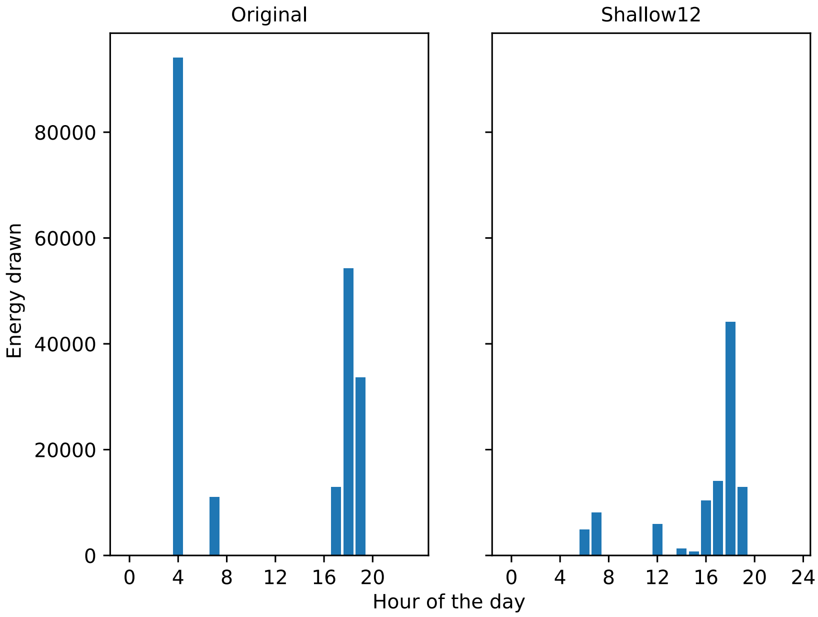

Figure 5.14: An example of poor reconstruction in the time range 4–5 in the microwave dataset.

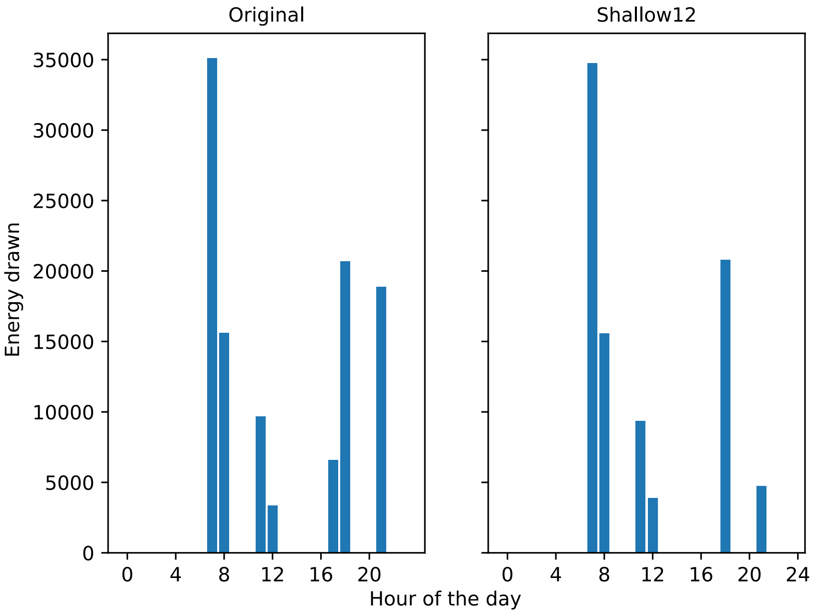

Figure 5.15: An example of poor reconstruction in time range 21–22 in the microwave dataset.

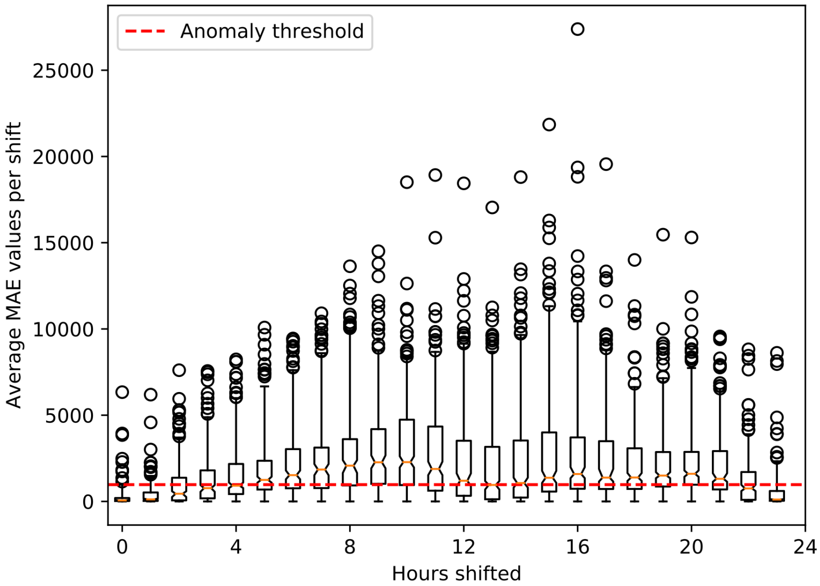

Figure 5.16: Box plots of the average reconstruction error per shift in the rolled datasets, obtained by the shallow model with bottleneck layer size of twelve.

the same as the ones for the box plots reported in Figure 5.8. It is possible to see that the overall shape of this plot remains similar to the TV plot in Figure 5.8, but with much lower MAE values. Overall, only the average MAEs of three shifts are completely below the anomaly threshold, compared with the almost six of the TV dataset. This can intuitively be explained by the fact that a mixture of three Gaussians, contrarily to a single one, allows on average more data points to not be exceedingly out of place. Therefore, the reconstruction errors are generally lower. From the box plots, it appears even more clearly how the interquartile ranges of the MAE values are closer to the anomaly threshold compared with the TV dataset. Looking into the shift with higher average MAE value, in position 10, approximately 26% of the days are *not* recognized as anomalies. The days not recognized as anomalous have very low values. This means that even if the reconstruction is actually pretty poor, spreading the errors over all the features allows the anomaly threshold not to be overcome. In any case, 26% is more than the 6.6% of the TV dataset for the shift with higher average MAE value. The amount of empty days in the two datasets is very close. Therefore, one can conclude that with a dataset–in which the average daily distribution of the power consumption approximates a mixture of Gaussians–the model is less sensitive to rolled anomalies; compared to a dataset in which the average daily distribution approximates a single Gaussian. This complicates detecting conditions like sundowning, or time shifts in the habits of the user. For this reason, it might be useful to investigate the adoption of another feature to train the model when the average daily distribution of the power consumption approximates a mixture of Gaussians.

Combining the rolled and scaled anomalies does not bring forth anything new compared to the analysis made for the TV dataset. Regarding the flattened anomalies, the behaviour of the models is exactly the same as with the TV dataset; therefore, there is nothing to add.

5.3 Considerations post-analysis

In the first part of this section, the hypotheses presented in Section 1.4 are discussed with respect to the new information gathered from the analysis of the results. Then, some of the limitations which emerged from the analysis of the models are listed.

Hypothesis 1: *"The smaller the bottleneck layer, the higher the number of anomalies found"*. With the current settings, this statement is *not* always true. For the original test set this statement holds: shallow12, shallow8, and shallow4 detect 16, 47, and 143 anomalies respectively. On the other hand, Figure 5.17 shows the number of anomalies detected in each shift of the rolled anomalies for all the shallow models. Previously, Figure 5.9 has illustrated the same data of one of these models, therefore all the comments regarding it also apply to Figure 5.17. The values are connected by a line to highlight the trend. As it can be seen, the number of anomalies detected by models with smaller latent representations is generally smaller than the number of anomalies detected by models with bigger latent representations. This counter-intuitive result is caused by the fact that models with smaller latent representation have higher thresholds for anomalies, as shown in Figure 5.1. Therefore, lowering those thresholds causes the number of anomalies to increase sharply, as it can be seen from the line labelled *mixed* in the graph. That line represents the number of anomalies that shallow4 would detect if it had the same anomaly thresholds that shallow12 has.

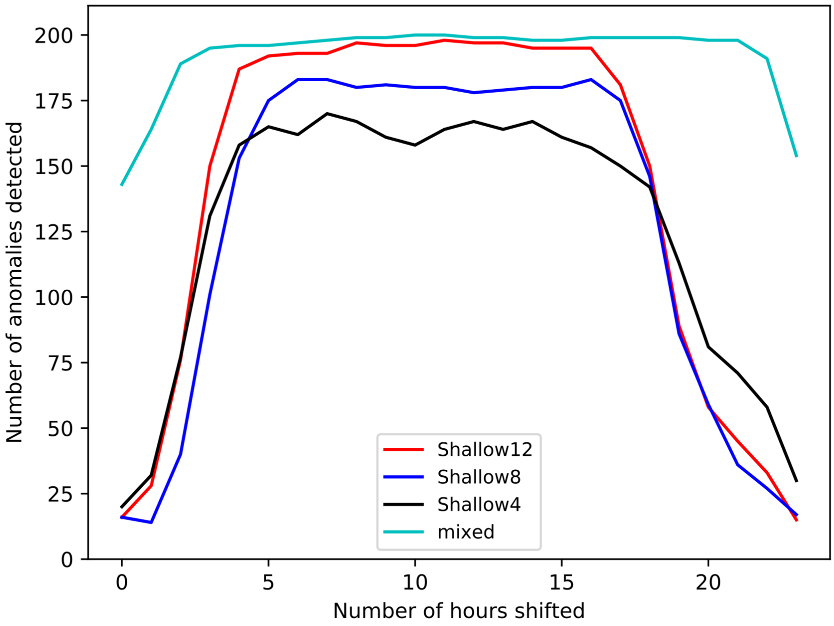

Figure 5.17: Number of anomalies detected per shift in the rolled datasets by each the shallow models.

Hypothesis 2: *"The anomalies found by models with bigger latent representations are a subset of the anomalies found by a model with a smaller latent representation"*. This is *not* true for the models trained during this research. As mentioned in the previous paragraph, sometimes models with smaller latent representations have a lower number of anomalies detected. Even if the thresholds were adjusted to counteract the increasing reconstruction error values, there is no guarantee that the anomalies of a model with smaller latent representation would be a perfect subset of the anomalies of a model with bigger latent representation. An example to support this hypothesis is shown in Figure 5.3, in which a model with smaller latent representation reconstructs an unusual input better than a model with a bigger latent representation. In general, depending on the rigidity of the model, a different notion of normality is learned. NNs weights are usually initialized with random values close to zero, drawn from a normal or uniform probability distribution. New methods that outperform this basic technique in specific contexts have been published, see for instance He et al. [69] for the RELU case. Nonetheless, an element of randomness is maintained in the weight initialization. Because of this intrinsic randomness in the initialization of the network, two models with identical hyperparameters might reach slightly different notions of normality. Therefore, models with similar but not identical hyperparameters do not necessarily reach notions of normality easily comparable. Hence, there is no reason according to which the anomalies of a model with smaller latent representation need to be a perfect subset of the anomalies of a model with bigger latent representation.

Hypothesis 3: *"the deep models are able to reach a more flexible learned representation compared to the shallow models. Therefore, the models identify as anomalies only the data points that heavily deviate from the norm"*. This statement is *partially* true. No substantial difference was found between the performance of the deep and shallow models. The representation learned by the deep model with bottleneck size of twelve was marginally more flexible than the representation learned by its shallow counterpart. Therefore, the deep model was able to reconstruct consumption in the night or early morning better, slightly decreasing the number of anomalies detected. Overall, the difference in performance between shallow and deep models was not meaningful.

During the analysis of the results, several limitations of the current model and approach have emerged and are reported below. Then, Section 6.2 proposes methods to overcome these limitations.

1. By shuffling the dataset before training the models, habits are considered as monolithic immutable entities. Therefore, the models do not exploit their temporal and mutable nature.

2. Days with zero or low consumption have low reconstruction errors. Therefore, they are never flagged as anomalies.

3. Choosing the 95$^{\text{th}}$ percentile of the training reconstruction error as fixed anomaly threshold cause models with small latent representation to not perform as expected.

4. The reconstruction error of solitary peaks in unusual times of the day is spread over all features. Therefore, it is not flagged as anomalous, especially in days with medium to low consumption.

5. Datasets in which the average daily distribution of power consumption approximates a mixture of Gaussian are less sensitive to rolled anomalies.

Chapter 6

Discussion

This chapter puts this research in the context of related literature, describing commonalities and differences with other works. First, Section 6.1 discusses the related literature. Then, Section 6.2 focuses on the limitations of the present research, highlighting future steps to take in order to improve the performance of the models.

6.1 Related work

Several researchers used autoencoders in anomaly detection contexts previously, and this research uses their ideas as a starting point. First, the main works in this area are presented on their own merits. Then, they are discussed and their approach is compared to the present research.

Japkowicz's PhD thesis focuses on outlier detection with autoencoders [25]. The author uses autoencoders to model the *normal* class. Then, defines a threshold in the reconstruction error to classify an object as anomalous. Japkowicz's work is based on the assumption that if the network is not able to properly reconstruct the input–i.e. the reconstruction error is large–then the input is not part of a common pattern in the data and can be considered an outlier. The autoencoder used has a hidden layer with fewer nodes compared to the input layer, and the threshold is set at a value that minimises both false positive and false negative classifications. Japkowicz tests the performance of her model on real-world datasets and compares them with a Multilayer Perceptron (MLP) trained on examples of both the known class and unknown classes.

Hawkins et al. [26] used a Replicator Neural Network, which is closely related to an autoencoder. The main difference is the activation function: instead of using a classic activation function such as sigmoid or hyperbolic tangent, they use the formula reported in Equation 6.1.

$$S(\theta) = \frac{1}{2} + \frac{1}{2(k-1)} \sum_{j=1}^{N-1} tanh[\alpha(\theta - \frac{j}{N})] \tag{6.1}$$

Where N is the number of steps of the staircase, α controls how steep is the transition between different levels, k is the number of the layer, and θ is the weighted sum of the inputs to the unit. For $k = 3$ and $\alpha = 100$, the function has been plotted in figure 6.1. The staircase activation function is important because it partitions continuously distributed data points into a number of discrete-valued vectors, allowing the network to naturally cluster data. In order to decide if a data point is an outlier, the authors define the Outlier Factor as the MSE over all the features. The authors do not define a threshold, and instead only rank the data points according to the Outlier Factor. Another paper written by the same authors [70] presents a comparison of the performance of their system with other commonly employed anomaly detection methods on publicly available datasets. Dau et al. [68] build on top of this paper employing a three-layer Replicator NN, opposed to a five-layer one. Dau et al. demonstrate that for the application at hand the three-layer Replicator NN performed as well and in some cases better than the original system. Additionally, the three-layer NN is faster to train. The authors chose the maximum reconstruction error obtained on the training set as the anomaly threshold.

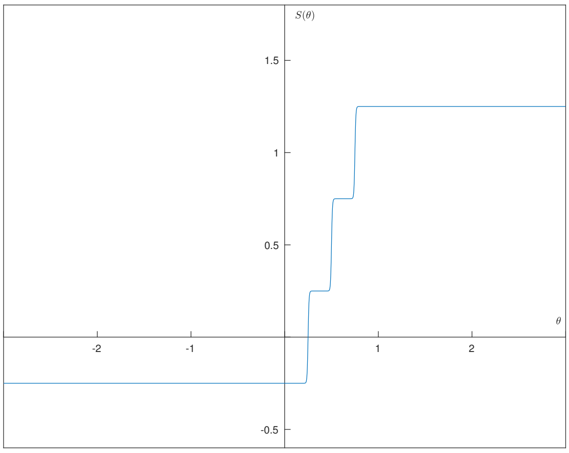

Figure 6.1: Staircase activation function.

Martinelli et al. [27] use autoencoders to detect anomalies in signals coming from substations of an electric power system. They use a sliding window to calculate two metrics that define if the current values are to be considered outliers. These two metrics are (1) the average reconstruction error over all the features and in all the time steps of the sliding window, and (2) the average of the absolute deviation of the features from their mean. A threshold for each measure is defined and, if overcome, an alarm is raised. The authors do not specify how the thresholds for the metrics were chosen.

Along the same lines, Sohn et al. [28] define a novelty index measuring the reconstruction error in terms of Euclidean distance. The data used in this work come from a simplified numerical model of a computer hard disk. Thompson et al. [29] use a very similar system, using as input data the average CPU utility over a period of 15 minutes in a network server. Both works do not explicitly define a threshold on the chosen metrics.

As it is possible to understand, all the works reported above, as well as the present research, work in the same way. They use autoencoders to create a compressed representation of the input data, they determine one or more metrics to judge the reconstruction of the input, and set thresholds on those metrics to classify data points as anomalous. The process of selecting the right threshold for the metrics is central, but most of the reported works either do not describe it or they bypass the need for it by ranking the data points. This process should be the subject of an investigation in its own right, in order to establish some guidelines for similar researches. Moreover, all of the aforementioned studies were not conducted recently. Therefore, they could not have taken into consideration later advancements in the field of NN such as denoising or variational autoencoders, or autoencoders based on recurrent NNs. The present thesis focused on understanding the basic building blocks of anomaly detection on appliance-level electrical data. Starting from these understandings several more sophisticated and more up-to-date solutions can be investigated. Some of these solutions are presented in Section 6.2.

Other works done in the general context of anomaly detection using autoencoders were considered, but since different approaches were adopted they are seen as not directly comparable with the present research. Erfani et al. [71] focus on high dimensional data and chose a hybrid approach: one-class SVM and Deep Belief Network. This NN is similar to an autoencoder, but not the same. Moreover, with 24 features the context of this thesis can be considered relatively low dimensional. Nicolau et al. [72] also use a hybrid approach, first training an autoencoder to reduce dimensionality and then analysing densities in the hidden layer, where a low density would indicate an anomaly. Their approach is similar to the one adopted in the thesis but the main difference is the choice of the datasets. The authors chose datasets in which the normal class is separated from the anomalous class, effectively framing the problem as a supervised learning problem. Sakurada et al. [73] compare the performance of autoencoders to linear PCA and kernel PCA, confirming that autoencoders generally perform better or require less computation than their counterpart. The authors used the aforementioned techniques to detect anomalies in both artificial and real data, but unfortunately not much detail is provided to assess the anomalies. Yan et al. [74] use stacked denoising autoencoders but just as an initial step to learn the most important features of the data. These learned features are then used as input for another classifier which is responsible for detecting the anomalies.

6.2 Future work

This section takes as a starting point the limitations of the current models listed in Section 5.3 and presents some ideas on how those limitations can be overcome, paving the way for future work. In addition, some other ideas to be investigated in the future are listed as well.

"By shuffling the dataset before training the models, habits are considered as monolithic immutable entities. Therefore, the models do not exploit their temporal and mutable nature." The current models do not take into consideration the sequential temporality of the data, but that is a key component of the dataset at hand. In future experiments, models that take into consideration this aspect should be examined. Autoencoders based on recurrent NNs are suited for this kind of task, and especially Long Short-Term Memory networks [75] should be analysed for the task at hand.

"Days with zero or low consumption have low reconstruction errors. Therefore, they are never flagged as anomalies." This is a problem, especially if the underlying assumption is that the monitored appliances should be used almost every day. In the scenario at hand, zero or low consumption could mean that something happened to the tenant, making it unable to use the appliance. Because of the relationship between low input values and the reconstruction errors mentioned in Section 5.1.1, this behaviour's correction is challenging. One solution could be to enforce a threshold on the number of consecutive days with no consumption allowed, but this is a very rigid option. A better version of this solution would be to learn a good value for this threshold from the training data, once anomalies and vacation days are removed from it. Another solution entails transforming the skewed distribution of the reconstruction errors (see Section 4.3), bringing it closer to a normal distribution. This can be done simply by taking the logarithm of the values or using a more complex technique such as the Box-Cox transformation [76]. Once the data distribution is closer to a normal, lower and upper thresholds can be set. In this case, the lower threshold would be useful to identify as anomalies the data points that are empty or with extremely low values.

"Choosing the 95th percentile of the training reconstruction error as fixed anomaly threshold cause models with small latent representation to not perform as expected." Analysing the results of the current models, it appeared that the smaller the size of the bottleneck layer, the bigger are the reconstruction errors, and therefore the bigger the values of the 95th percentile of the reconstruction errors of the training set. This is not an ideal behaviour, thus another method to determine the threshold for shrinking latent

representations should be devised. Ideally, the thresholds are determined dynamically according to the size of the bottleneck layer to account for a bigger average error reconstruction.

"The reconstruction error of solitary peaks in unusual times of the day is spread over all features. Therefore, it is not flagged as anomalous, especially in days with medium to low consumption." This happens because, when calculating the MADEs, the deviation of each feature's reconstruction error from the mean is averaged over all features. Therefore, to identify days in which the reconstruction error of only one feature is extremely high and of the others is low, one should add another metric that does not average their values.

"Datasets in which the average daily distribution of power consumption approximates a mixture of Gaussian are less sensitive to rolled anomalies." This makes it more difficult to detect potentially dangerous shifts in the habits of the tenant. This problem is intrinsic to the kind of usage a person does of the appliance and is not easily solved. Stricter anomaly thresholds might help, as might also a model with a stronger notion of temporality.

Another option that could improve performance is to consider other features to learn. The sum of the power values per hour seemed a sensible and straightforward feature, but some other feature can possibly be investigated. One of the deep learning advantages over more traditional machine learning techniques is its ability to autonomously select good features from raw data. Therefore, another investigation direction is to use directly the raw power data as input to the models. This might translate into more preprocessing work on the dataset to make sure that all data points needed are in place for all considered appliances. However, it might improve the performance of the model overall.

Finally, something that can definitely improve the performance of the models is a bigger hyperparameter search space. Together with all the variables typically considered in this search, different optimizers and parameters per optimizer should be tested.

Chapter 7

Conclusion

This thesis discusses how an autoencoder can be trained to identify anomalies in usage patterns of appliance-level power data. Furthermore, it explores the impact of important hyperparameters such as the architecture of the NN, different node activation functions, and loss functions on the resulting models. Several choices were made, both for the definition of anomalies in the context at hand, and their evaluation. Chapters 5 and 6 highlight the limitations of the current models and propose improvements to overcome them. The answers to the research questions are reported below.

Sub-question 1

"What kinds of anomaly are the models able to recognize?"
If the average daily usage of an appliance approximated a single Gaussian distribution, the trained autoencoders were very sensitive to rolled anomalies. On the other hand, if the average daily usage of an appliance approximated a mixture of Gaussians, the model showed less sensitivity. Furthermore, because of the linear relationship existing between the input scale and the reconstruction errors, scaling factors have to be consistently bigger than 1 for the model to effectively recognize the input as anomalous. This fact also explains why inputs with exclusively low values are never recognized as anomalies. The models are also very sensitive to the flattened anomalies, as long as the flattened value is not very low.

Sub-question 2

"How does performance evolve when the size of the latent representation is increasingly reduced?"
As expected, smaller latent representations lead to more rigid models which learn only increasingly common usage patterns. On the other hand, what was not expected was the large role that the selection of the anomaly thresholds plays. Thresholds should be defined dynamically taking into account the

rigidity of the model, simply choosing the 95% percentile of the training errors does not suffice.

Sub-question 3

"Is there a clear performance improvement when a deep autoencoder is used instead of a shallow autoencoder?"
Compared to their shallow counterparts, deeper autoencoders learn a more flexible representation of the training data and are able to adapt better to it. The difference in reconstruction error between a deeper autoencoder and their shallow counterparts is not huge, but the former performs better. This is not necessarily a good sign for anomaly detection, because to detect anomalies the right amount of rigidity is needed. Because of this, and because the deeper version takes substantially longer to train than the shallow version, deeper autoencoders are deemed less appropriate for the current task and the current data.

As previously mentioned, the first goal of AAL is defined as "extending the time people can live in their preferred environment by increasing their autonomy, self-confidence and mobility". In this context, the present research investigated the use of autoencoders as anomaly detection tools for appliance-level electrical data. As long as the average power consumption approximates a normal distribution, autoencoders are sensitive to most of the anomalies described in Section 3.4. The issue of empty or low valued input vectors can be improved as suggested in Section 6.2. It follows that, for this kind of input, a system based on autoencoders could serve as first care solution to detect a major divergence from normal behaviour. The models trained in this research were intended to investigate the general capabilities of autoencoders to detect anomalies. Therefore, a system deployed on premises may need to adopt more sophisticated solutions. For instance, the current models take as input a vector which represents the consumption of a day starting from midnight. This choice is arbitrary; depending on the context, a smaller time frame or a different hour interval might be needed. Aside from practical details, the underlying autoencoder technique might be successfully used for anomaly detection of power consumption.

To conclude, the basic autoencoder model is able to detect anomalies in usage patterns of appliance-level power data. Its effectiveness depends mostly on the type of anomalies and the average distribution of the input data. The model has several limitations, listed in Section 5.3, but extending its capabilities as proposed in Section 6.2 will no doubt improve its performance. Overall, this work analysed important aspects of the basic autoencoder model for anomaly detection; several strengths of the model were found, as well as weaknesses. Building on this knowledge, if the proposed improvements are implemented, better performance will be achieved .

Appendix A

Implementation details

The system is implemented with the Python programming language because of its scientific computation libraries and user-friendliness. Keras[1] has been chosen as the NN API because of its focus on enabling fast experimentation. Moreover, Keras can be used as a front-end for other NN APIs such as TensorFlow and Theano, masking their complexity and speeding up the development process. All data manipulation and plots generation was performed using Python libraries as well. The code of the thesis project is publicly hosted on BitBucket[2]. The development process was executed on an Apple MacBook Pro and its main hardware components can be found in Table A.1a, while the main software and libraries used are listed in Table A.1b.

Component	**Specification**
CPU	Intel Core i5-4278U
RAM	8GB, 1600 MHz, DDR3L
Hard Drive	APPLE SSD SM0256F

(a) Hardware components

Software	**Version**
Mac OS X	10.11.6
Python	3.6.4
Keras	2.1.4
Tensorflow	1.5.0
Numpy	1.14.0
Pandas	0.22.0
Matplotlib	2.1.2

(b) Software used

Table A.1: Hardware and software specifications

[1] `https://keras.io`

[2] `https://bitbucket.org/RiccB/thesis_project`

Bibliography

[1] Eurostat. (18th May 2018). Old-age-dependency ratio, [Online]. Available: http://ec.europa.eu/eurostat/web/products-datasets/-/tps00198 (visited on 23/06/2018).

[2] Eurostat. (18th May 2018). Projected old-age dependency ratio, [Online]. Available: http://ec.europa.eu/eurostat/web/products-datasets/-/tps00200 (visited on 23/06/2018).

[3] N. Triggle, 'One million older people in need "struggle alone" ', *BBC*, 21st Oct. 2015. [Online]. Available: http://www.bbc.com/news/health-34576012 (visited on 23/06/2018).

[4] H. Sun, V. De Florio, N. Gui and C. Blondia, 'Promises and challenges of ambient assisted living systems', in *Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on*, Ieee, 2009, pp. 1201–1207.

[5] AAL-EU. (2018). Active and assisted living programme, [Online]. Available: http://www.aal-europe.eu/about/objectives/ (visited on 23/06/2018).

[6] P. Rashidi and A. Mihailidis, 'A survey on ambient-assisted living tools for older adults', *IEEE journal of biomedical and health informatics*, vol. 17, no. 3, pp. 579–590, 2013.

[7] N. Noury, M. Berenguer, H. Teyssier, M.-J. Bouzid and M. Giordani, 'Building an index of activity of inhabitants from their activity on the residential electrical power line', *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 5, pp. 758–766, 2011.

[8] X. Zhang, T. Kato and T. Matsuyama, 'Learning a context-aware personal model of appliance usage patterns in smart home', in *Innovative Smart Grid Technologies-Asia (ISGT Asia)*, IEEE, 2014, pp. 73–78.

[9] V. Chandola, A. Banerjee and V. Kumar, 'Anomaly detection: A survey', *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[10] V. Hodge and J. Austin, 'A survey of outlier detection methodologies', *Artificial intelligence review*, vol. 22, no. 2, pp. 85–126, 2004.

[11] V. Chandola, A. Banerjee and V. Kumar, 'Outlier detection: A survey', *ACM Computing Surveys*, 2007.

[12] M. Markou and S. Singh, 'Novelty detection: A review–part 1: Statistical approaches', *Signal Processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[13] M. Markou and S. Singh, 'Novelty detection: A review–part 2: Neural network based approaches', *Signal Processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[14] Y. Zhang, W. Chen and J. Black, 'Anomaly detection in premise energy consumption data', in *Power and energy society general meeting, 2011 ieee*, IEEE, 2011, pp. 1–8.

[15] C. Nordahl, M. Persson and H. Grahn, 'Detection of residents' abnormal behaviour by analysing energy consumption of individual households', in *IEEE International Conference on Data Mining series (ICDM), New Orleans*, IEEE, 2017, pp. 729–738.

[16] 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.

[17] G. W. Hart, 'Residential energy monitoring and computerized surveillance via utility power flows', *IEEE Technology and Society Magazine*, vol. 8, no. 2, pp. 12–16, 1989.

[18] J. Alcalá, J. Ureña and Á. Hernández, 'Activity supervision tool using non-intrusive load monitoring systems', in *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*, IEEE, 2015, pp. 1–4.

[19] J. Alcalá, O. Parson and A. Rogers, 'Detecting anomalies in activities of daily living of elderly residents via energy disaggregation and cox processes', in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, ACM, 2015, pp. 225–234.

[20] J. M. Alcalá, J. U. Urena, A. Hernandez and D. Gualda, 'Sustainable homecare monitoring system by sensing electricity data', *IEEE Sensors Journal*, 2017.

[21] J. M. Alcalá, J. Ureña, Á. Hernández and D. Gualda, 'Assessing human activity in elderly people using non-intrusive load monitoring', *Sensors*, vol. 17, no. 2, p. 351, 2017.

[22] D. R. Cox, 'Some statistical methods connected with series of events', *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 129–164, 1955.

[23] A. P. Dempster, ‘Upper and lower probabilities induced by a multivalued mapping’, *The annals of mathematical statistics*, pp. 325–339, 1967.

[24] G. Shafer *et al.*, *A mathematical theory of evidence.* Princeton university press Princeton, 1976, vol. 1.

[25] N. Japkowicz, *Concept-learning in the absence of counter-examples: an autoassociation-based approach to classification.* 1999.

[26] S. Hawkins, H. He, G. Williams and R. Baxter, ‘Outlier detection using replicator neural networks’, in *International Conference on Data Warehousing and Knowledge Discovery*, Springer, 2002, pp. 170–180.

[27] M. Martinelli, E. Tronci, G. Dipoppa and C. Balducelli, ‘Electric power system anomaly detection using neural networks’, in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, Springer, 2004, pp. 1242–1248.

[28] H. Sohn, K. Worden and C. Farrar, ‘Novelty detection using auto-associative neural network’, Los Alamos National Lab., NM (US), Tech. Rep., 2001.

[29] B. B. Thompson, R. J. Marks, J. J. Choi, M. A. El-Sharkawi, M.-Y. Huang and C. Bunje, ‘Implicit learning in autoencoder novelty assessment’, in *Neural Networks, 2002. IJCNN’02. Proceedings of the 2002 International Joint Conference on*, IEEE, vol. 3, 2002, pp. 2878–2883.

[30] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006, ISBN: 0387310738.

[31] D. M. J. Tax, ‘One-class classification’, 2001.

[32] L. Tarassenko, P. Hayton, N. Cerneaz and M. Brady, ‘Novelty detection for the identification of masses in mammograms’, 1995.

[33] V. Barnett and T. Lewis, *Outliers in statistical data.* Wiley, 1974.

[34] E. Parzen, ‘On estimation of a probability density function and mode’, *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[35] V. Vapnik, *Statistical learning theory. 1998.* Wiley, New York, 1998.

[36] T. Kohonen, ‘The self-organizing map’, *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[37] I. Goodfellow, Y. Bengio, A. Courville and Y. Bengio, *Deep learning.* MIT press Cambridge, 2016, vol. 1.

[38] D. E. Rumelhart, G. E. Hinton and R. J. Williams, ‘Learning internal representations by error propagation’, California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[39] H. Bourlard and Y. Kamp, 'Auto-association by multilayer perceptrons and singular value decomposition', *Biological cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[40] G. E. Hinton and R. S. Zemel, 'Autoencoders, minimum description length and helmholtz free energy', in *Advances in neural information processing systems*, 1994, pp. 3–10.

[41] G. W. Cottrell and P. Munro, 'Principal components analysis of images via back propagation', in *Visual Communications and Image Processing'88: Third in a Series*, International Society for Optics and Photonics, vol. 1001, 1988, pp. 1070–1078.

[42] P. Baldi and K. Hornik, 'Neural networks and principal component analysis: Learning from examples without local minima', *Neural networks*, vol. 2, no. 1, pp. 53–58, 1989.

[43] N. Japkowicz, S. J. Hanson and M. A. Gluck, 'Nonlinear autoassociation is not equivalent to pca', *Neural computation*, vol. 12, no. 3, pp. 531–545, 2000.

[44] G. E. Hinton and R. R. Salakhutdinov, 'Reducing the dimensionality of data with neural networks', *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[45] I. H. Witten, E. Frank, M. A. Hall and C. J. Pal, *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2016.

[46] C. Poultney, S. Chopra, Y. L. Cun *et al.*, 'Efficient learning of sparse representations with an energy-based model', in *Advances in neural information processing systems*, 2007, pp. 1137–1144.

[47] Y.-l. Boureau, Y. L. Cun *et al.*, 'Sparse feature learning for deep belief networks', in *Advances in neural information processing systems*, 2008, pp. 1185–1192.

[48] A. Makhzani and B. Frey, 'K-sparse autoencoders', *arXiv preprint arXiv:1312.5663*, 2013.

[49] P. Vincent, H. Larochelle, Y. Bengio and P.-A. Manzagol, 'Extracting and composing robust features with denoising autoencoders', in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 1096–1103.

[50] Y. Bengio, P. Lamblin, D. Popovici and H. Larochelle, 'Greedy layer-wise training of deep networks', in *Advances in neural information processing systems*, 2007, pp. 153–160.

[51] H. Larochelle, D. Erhan, A. Courville, J. Bergstra and Y. Bengio, 'An empirical evaluation of deep architectures on problems with many factors of variation', in *Proceedings of the 24th international conference on Machine learning*, ACM, 2007, pp. 473–480.

[52] Y. Bengio, L. Yao, G. Alain and P. Vincent, 'Generalized denoising auto-encoders as generative models', in *Advances in Neural Information Processing Systems*, 2013, pp. 899–907.

[53] G. Alain and Y. Bengio, 'What regularized auto-encoders learn from the data-generating distribution', *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3563–3593, 2014.

[54] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent and S. Bengio, 'Why does unsupervised pre-training help deep learning?', *Journal of Machine Learning Research*, vol. 11, no. Feb, pp. 625–660, 2010.

[55] D. P. Kingma and M. Welling, 'Auto-encoding variational bayes', *arXiv preprint arXiv:1312.6114*, 2013.

[56] N. Srivastava, E. Mansimov and R. Salakhudinov, 'Unsupervised learning of video representations using lstms', in *International conference on machine learning*, 2015, pp. 843–852.

[57] J. Masci, U. Meier, D. Cireşan and J. Schmidhuber, 'Stacked convolutional auto-encoders for hierarchical feature extraction', in *International Conference on Artificial Neural Networks*, Springer, 2011, pp. 52–59.

[58] S. Rifai, P. Vincent, X. Muller, X. Glorot and Y. Bengio, 'Contractive auto-encoders: Explicit invariance during feature extraction', in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, Omnipress, 2011, pp. 833–840.

[59] W. Kleiminger, C. Beckel and S. Santini, 'Household occupancy monitoring using electricity meters', in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2015)*, Osaka, Japan, Sep. 2015.

[60] A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro and A. M. Tonello, 'Greend: An energy consumption dataset of households in italy and austria', in *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, IEEE, 2014, pp. 511–516.

[61] Department of Energy and Climate Change and Intertek Testing and Certification Ltd., *Household electricity survey, 2010-2011*, `http://doi.org/10.5255/UKDA-SN-7874-1`, UK Data Service, 2016.

[62] J. Kelly and W. Knottenbelt, 'The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes', *Scientific Data*, vol. 2, no. 150007, DOI: `10.1038/sdata.2015.7`.

[63] L. K. Evans, 'Sundown syndrome in institutionalized elderly', *Journal of the American Geriatrics Society*, vol. 35, no. 2, pp. 101–108, 1987.

[64] X. Glorot, A. Bordes and Y. Bengio, 'Deep sparse rectifier neural networks', in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[65] Stanford University. (2018). Convolutional neural networks for visual recognition, [Online]. Available: `http://cs231n.github.io/neural-networks-3/` (visited on 23/06/2018).

[66] S. Ruder, 'An overview of gradient descent optimization algorithms', *arXiv preprint arXiv:1609.04747*, 2016.

[67] D. P. Kingma and J. Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, 2014.

[68] A. H. Dau, V. Ciesielski and A. Song, 'Anomaly detection using replicator neural networks trained on examples of one class.', in *SEAL*, 2014, pp. 311–322.

[69] K. He, X. Zhang, S. Ren and J. Sun, 'Delving deep into rectifiers: Surpassing human-level performance on imagenet classification', in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[70] G. Williams, R. Baxter, H. He, S. Hawkins and L. Gu, 'A comparative study of rnn for outlier detection in data mining', in *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, IEEE, 2002, pp. 709–712.

[71] S. M. Erfani, S. Rajasegarar, S. Karunasekera and C. Leckie, 'High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning', *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[72] M. Nicolau, J. McDermott *et al.*, 'A hybrid autoencoder and density estimation model for anomaly detection', in *International Conference on Parallel Problem Solving from Nature*, Springer, 2016, pp. 717–726.

[73] M. Sakurada and T. Yairi, 'Anomaly detection using autoencoders with nonlinear dimensionality reduction', in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, ACM, 2014, p. 4.

[74] W. Yan and L. Yu, 'On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach', in *Proceedings of the annual conference of the prognostics and health management society*, 2015.

[75] S. Hochreiter and J. Schmidhuber, 'Long short-term memory', *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[76] G. E. Box and D. R. Cox, 'An analysis of transformations', *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 211–252, 1964.