UNIVERSIDAD PONTIFICIA COMILLAS
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

PROYECTO FIN DE MASTER

PROBABILISTIC RESIDENTIAL LOAD FORECASTING BASED ON LSTM RECURRENT NEURAL NETWORKS: IMPLEMENTATION AND ASSESSMENT

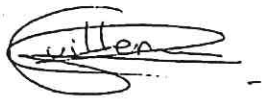
AUTOR: Guillermo Moraleda Conejo

DIRECTOR: Baptiste Feron

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Probabilistic Residential Load Forecasting based on LSTM Recurrent Neural Networks: Implementation and Assessment

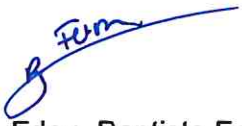
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2017-2018 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: Guillermo Moraleda Conejo Fecha: 05/06/2018

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Baptiste Feron Fecha: 06/06/2018

Abstract

The electric power industry is shifting towards more intelligent, flexible, and interactive systems with an increasing share of renewable energy generation. At a household level this is achieved through household energy management systems HEMS that benefit from smart meter technology. However, HEMS are only effective if reliable household electrical load forecasts are generated. Due to its fluctuating nature, forecasting future electrical loads has proven to be a challenging task. In this work probabilistic forecasting methods are proposed, unlike deterministic forecasts, probabilistic forecasts provide further information on future load values distributions. Long short-term memory LSTM recurrent neural network RNN models are proposed to improve the forecasting accuracy; these models have proven to be effective when forecasting volatile time series data. In the first part of this thesis the forecasting results from the LSTM RNN models are compared against the more commonly used feedforward neural network models. Both models are evaluated on four smart meter data sets, Smart*, UCI, UK-DALE and Ausgrid data set. The main finding is that although both models generate reliable density forecasts, the LSTM RNN outperforms the feedforward neural network on most of the cases. Forecasting models perform differently depending on the data set under consideration. The second part of this thesis seeks to extract the main data set features affecting both model's forecasting accuracy. A correlation study of the forecasting errors and each feature's metric was conducted. The 300-household data from the Ausgrid data set was analyzed to generate the correlation analysis. Results showed that for the analyzed households, the average consumption power, the cyclic behavior, the load dispersion, and the behavior pattern consistency impact forecasting accuracy.

Keywords: STLF, Household, Recurrent Neural Network, Probabilist Forecasting, Load Pattern, Clustering

Contents

1 Introduction

The energy market is facing a paradigm shift. The electric power generation is now moving from fossil fuels to renewable sources and from centralized to decentralized power generation. The fluctuating nature of wind and solar electric power generation is a big challenge when adopting these technologies. In fact, the spread of renewable power generation has affected wholesale electricity markets by increasing their price volatility. On the other hand, there is a current spread on the employment of smart meters on households. Smart meter technology enables an implementation of variable rate electricity tariffs which are adjusted to the wholesale energy market.

A household energy management system (HEMS) is designed to optimize consumer costs by managing consumer demand, communicating with the utility to determine the wholesale electricity market price, and calculating the use of local generation to off-set remote generation costs [32]. To achieve the aforementioned objectives, HEMS require reliable forecasts of short-term electricity consumption. This work focuses on developing better short-term load forecasts by using machine learning algorithms.

Although most research studies have focused on developing single point load forecasts, in this work a probabilistic electricity load forecast is proposed. Household electricity consumption depend largely on arbitrary human behavior, this causes high forecasting errors. Probabilistic forecasts provide further information on the distribution of future values.

In this work a forecasting method based on Long Short Term Memory (LSTM) recurrent neural networks is proposed. Unlike feedforward artificial neural networks, LSTM networks present a memory cell that enables establishing long-term dependencies in the data. As human behavior is thought to be a main factor of influence on household electricity consumption, extracting complex consumption patterns within the household data could considerably improve the forecasting performance.

Furthermore, the forecasting accuracy largely depends on the data set under consideration. Selecting the optimal forecasting method can be difficult due to the disperse results obtained for different smart meter data sets. This thesis conducts an impact study to analyze the main data set factors of influence on forecasting performance. The selected factors extract not only simple data set features such as the data set dispersion or average value, but they also seek to extract household behavioral patterns.

First, the *Literature Review* analyzes the existing literature on household probabilistic forecasting methods, household data characterization and presents publicly available smart meter data sets used in research studies.

Second, a theoretical description of forecasting methods and an explanation of time series data characterization are presented in the *Method Fundamentals* chapter.

Third, the *Implementation* chapter provides a description of the selected data sets, the forecasting model set up and the software modules developed to implement all methods.

Finally, the *Assessment* chapter compares the forecasting performance of the selected forecasting methods and presents the results from the data set features impact study.

2 Literature Review

This chapter provides a literature review on probabilistic short-term load forecasting on a single household level. In this chapter, the most common approaches to predict household loads are presented, the main challenges when forecasting residential loads are analyzed and the current trends within the research community are presented. The second part of the chapter analyzes the existing research on extracting data features affecting time series forecasting. The last section specifies the selected methodologies followed in this work and state the main contributions expected from the results.

Individual household loads are fundamentally different from aggregated ones [38]. On an aggregated level, load profiles tend to be smoother and easier to predict. The stochastic behavior of individual households is no longer present at high aggregation levels [24, 28]. Furthermore, on high levels of aggregation, the variations on the load profiles have high correlation with external variables such as temperature, calendar or sunlight radiant flux [27].

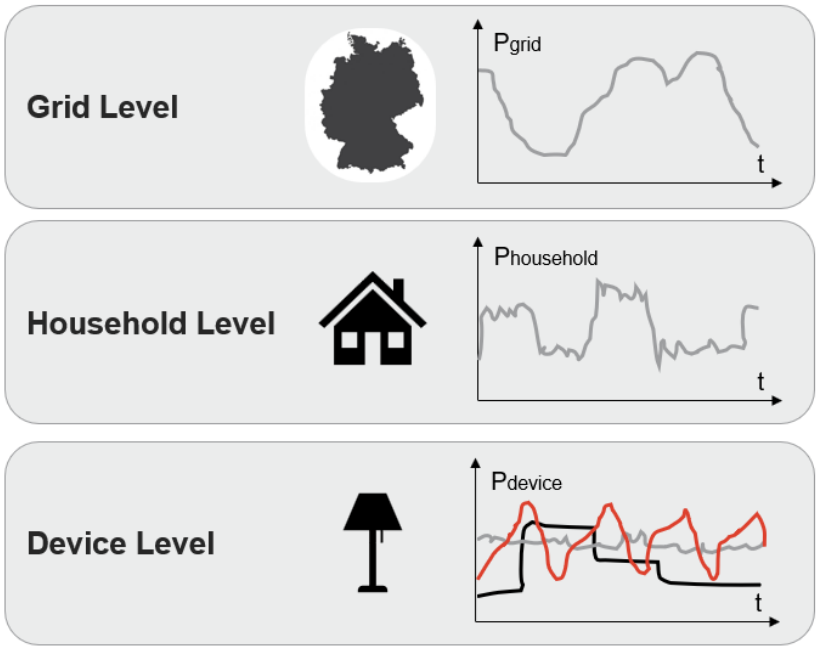


Figure 2.1: Typical load profiles at three different levels of aggregation

There is no overall accepted standard for classifying the range of load forecasts. However, a commonly used approach consits on grouping forecasting methods in four

categories: very short term load forecasting (VSTLF), short term load forecasting (STLF), medium term load forecasting (MTLF), and long term load forecasting (LTLF). The usually accepted cut-off time horizons are one day, two weeks, and three years respectively [20]. A more general classification may lead to two categories: short term load forecasting (STLF) for horizons shorter than two weeks and long term load forecasting (LTLF) for longer horizons [10]. Throughout this thesis the term STLF refers to a forecast horizon of one hour.

Smart meter data is highly volatile and forecasting point values does not provide meaningful information about the uncertainty of future demand [38]. Probabilistic load forecasting PLF embeds more information that enable users to make scenario-based decisions. PLFs are usually represented as quantiles [38], intervals or probability density functions [43]. The existing literature on short-term PLFs on a single household level is very limited. Other fields such as probabilistic forecasting, or probabilistic short term load forecasting on an aggregated level have a much broader literature [37]. To the author's knowledge, apart from electric load forecasting, probabilistic forecasting is commonly used in hydrology [26], stock market [35], and wind speed forecasting [44].

The following sections present a review of the most commonly used techniques and methodologies to develop probabilistic short term electrical demand forecasting on a single household level. As done in [37], we will use the word "technique" to refer to a group of models that fall in the same family and "methodology" to represent a general solution framework that can be implemented with multiple techniques.

2.1 Probabilistic forecasting methods for residential load forecasting

Probabilistic forecasting for smart meter data is a very recent topic, there is sparse literature available in this field. To the best of the authors knowledge, there are three publications regarding this topic [1, 38, 13]. Furthermore, [43] analyzes the implementation of ANNs for household probabilistic forecasting. In table 2.1 the main characteristics on the three publications regarding probabilistic load forecasting at a household level are shown.

[1] considered kernel density estimation methods to generate probabilistic forecasts for 800 residential consumers and 200 small to medium-size enterprises, the developed models outperformed the unconditional KD estimator.

[38] developed forecasting models both at an aggregated and at a disaggregated level. It was shown that at a disaggregated level, the quantile forecasts outperform forecasts based on a normal distribution. Normal forecasts produce predictive densities which are too concentrated, not matching the true uncertainty. [13] developed and employed its forecasting model in 226 individual buildings. The forecasting model outperformed the persistence method results by 15%.

Due to the low number of publications on probabilistic electrical load forecasting for individual households, it is necessary to analyze the existing the literature on deterministic STLF on households. These two forecasting methods analyze the same type of data which support the assumption that the best techniques for deterministic forecasting should also perform well in probabilistic STLF.

Load forecasting techniques are typically classified into two groups [37]: statistical techniques and artificial intelligence techniques. The most-commonly used statistical techniques for STLF are: semi-parametric models, autoregressive and moving average (ARMA) models, and exponential smoothing models. On the contrary, the most commonly used artificial intelligence techniques for STLF are: artificial neural networks (ANN), fuzzy regression models, support vector machines (SVMs), and gradient boosting machines.

Statistical techniques are the traditionally used approach for load forecasting at an aggregated level. These techniques are based on time series or regression analysis and largely rely on the correlation between the predicted load and its past values. By statistical techniques we refer to all models that use conventional statistical methods to develop forecasts.

[33] compared statistical and artificial intelligence techniques for deterministic STLF on a household level. Autoregressive moving average (ARMA) and autoregressive (AR) methods were compared with support vector machine (SVM). SVM outperformed ARMA and AR methods on most cases, however, for some cases the developed models were worse than the persistence method used as a benchmark.

In [14] a spectral analysis and Kalman filtering method was employed for deterministic STLF on households. In this method, the deterministic component modelling the household lifestyle and a gaussian noise signal modelling the weather-dependent component are first added, then a Kalman filter is used to predict the residential load for different sampling periods and forecasting horizons. The results from this publication are not compared with standard benchmarks, therefore no further conclusion about its results can be extracted.

Although statistical techniques are reliable when predicting residential loads at an aggregated level [37, 11, 7], single household data has no-linearities and dynamic fluctuations that make the forecasting much more difficult [33, 27]. Statistical techniques fail to extract behavioral pattern from data. Therefore, *Artificial Intelligence* techniques are proposed because of their ability to extract features within the data with a higher degree of abstraction [33, 24, 28].

Artificial Intelligence AI techniques make use of *machine learning* fundamentals to develop load forecasts. The most significant artificial intelligence techniques for short term load forecasting on a single household level are support vector machine for regression (SVM) and artificial neural networks ANN. Artificial neural networks (ANN) is now the most commonly-used technique for developing deterministic STLF for residential buildings [2, 29, 12, 24, 28, 43]

The Long Short Term (LSTM) Recurrent Neural Network (RNN) follow the same principles as the feedforward ANN but it enables a higher degree of abstraction. It

was originally introduced in [19]. It initially received an important attention in the field of sequence learning.

The LSTM recurrent neural network have better capability of learning temporal correlations than feedfoward ANNs, this has been proved in fields such as speech recognition [16], image captioning [41], and language translation [36]. Furthermore, LSTM RNNs have also improved time series forecasting in the stock market prediction [8], wind speed prediction [3] and sea temperature prediction [45].

Single family loads present temporal correlations based on its resident's behavior that are hard to learn. Therefore, LSTM RNNs have been proposed as a method capable of extracting more complex temporal correlations. [24, 28] are examples where LSTM have proven to improve forecasting accuracy in comparison to ANN models. To the author's knowledge, there is no publication related to the application of LSTM RNN to probabilistic STLF for individual households.

The *persistence method* is a frequently used benchmark for STLF [33, 43, 27]. It is the simplest forecasting method, where forecasts equal the last observation. For example, for one hour ahead forecasting the resulting forecast will be $\hat{y}_{t-1} = y_t$. This method takes advantage of the fact that the load remains relatively constant for a short period. Literature shows that the *persistence method* is hard to beat in short-term load forecasting [33, 27].

Lastly, there are multiple metrics that evaluate forecasting accuracy. For deterministic forecasts the most commonly used metrics are the mean absolute error (MAE)[43] that average the forecasting error among samples, the mean absolute percentage error (MAPE) that give the error value as a percentage [24], and the root mean squared error RMSE [28].

There are also multiple metrics that evaluate the forecasting accuracy of probabilistic STLF. Due to its simple application, the continuous ranked probability score (CRPS) is the most commonly used method for evaluating these models [43, 1, 38]. However, in [37] most complex methods such as the *pinball loss* function or the *Winkler* method are proposed as more complex methods.

In this work, artificial neural networks (ANNs) and LSTM recurrent neural networks (RNNs) forecasting results will be compared with the *persistence method* to analyze their forecasting capability. To the author's knowledge, [43] is the only research work in which ANNs were employed for probabilistic household STLF. Furthermore, no research studies on the LSTM method for probabilistic household STLF was found. However, LSTM have been successfully employed in deterministic household STLF [24, 25, 28]. Mean absolute error (MAE) metric will be used to compare the developed models with the *persistence method*, while continuous ranked probability score (CRPS) will assess the performance of the probabilistic forecast.

2.2 Household load data characterization

Short term forecasting accuracy results tend to change importantly depending on the used data set [33, 43]. Different forecasting methods can perform very differently

depending on the data set employed. Therefore, it is important to study which data set characteristics cause these accuracy changes and what models can be used given specific data set characteristics. [31] analyzed the impact of data set features on different forecasting methods for time series data. In [24, 33] it is concluded that human behavioral patterns have an important impact on STLF of residential electricity consumption.

This work will intend to identify the main time series data determinants of forecasting accuracy, this means analyzing the impact of data characteristics on forecasting accuracy results. To do so, the most relevant features for time series data characterization will be selected. On the other hand, behavioral patterns will be identified and the impact of these patterns on the forecasting accuracy will also be studied.

[31] identified the main data features that determine a time series forecasting accuracy. Through simulations and empirical investigations, the impact of seven time series features (seasonality, trend, cycle, randomness, number of observations, inter-demand interval and coefficient of variation) and one strategic decision (forecasting horizon) on forecasting accuracy were analyzed. As [31] demonstrates, data cyclic patterns and data randomness strongly affect parametric approaches.

From the features analyzed in [31] trend, number of observations, inter-demand interval and forecasting horizon analysis will be discarded. The time lengths to be analyzed are less than one year and therefore no data trend behavior is expected. The number of observations, inter-demand interval and forecasting horizon will be kept constant within all datasets to focus the analysis on other features of greater interest to this work.

As stated in [24, 33] human behavioral patterns can be seen as a major factor of influence on the variation of household's electrical consumption. Therefore, the complexity and consistency of household electricity load patterns will be analyzed. Clustering household daily load profiles will extract these features.

The complexity of profile patterns refers to the number of well-defined patterns within a household. As an example, a household might have different daily routines during weekdays and weekends which would mean that there will be at least two well differentiated pattern profiles. The complexity of profile patterns will be measured by the size of the cluster with the highest number of daily profiles.

The randomness of household load profiles will be analyzed in terms of daily pattern consistency. Pattern consistency refers to the extent to which a household consumption behavior reflects defined patterns. As stated in [24] it is expected that the forecasting accuracy will diminish as the data consistency decreases. As done in [24], the pattern consistency will be defined by the number of outliers, which are daily profiles that do not correspond to any cluster of daily profiles.

The data features to be analyzed within this thesis are: average electrical load, seasonality, cycle, variation, behavioral pattern complexity and behavioral pattern consistency. These features have been selected to fulfill two objectives. The first one is to understand which factors affect the forecasting accuracy. The second objective

is to extract the household pattern behavior from the data and analyze the impact of these patterns on forecasting accuracy.

2.3 Data set selection

The broad smart meter deployment over the last decade has provided the industry with a big amount of data. There are multiple publicly available smart meter data sets with many different characteristics. Some of the most frequently used data sets used for household load forecasting are shown in table 5.2.

Name	Granularity	Recording period	Location	Households	Ref.
SGSC	30 min	48 months	Australia	10,000	[24]
CER	30 min	18 months	Ireland	782	[33]
Smart*	1 min	3 months	MA, USA	3	[4]
UCI	1 min	47 months	France	1	[39]
UK-DALE	6 s	5 - 21 months	UK	3	[22]
Ausgrid	30 min	12 months	Australia	300	[27]

Table 2.2: Data sets employed in research studies

The forecasting accuracy highly rely on the employed data set [33], therefore more than one data set are analyzed to extract the robustness of the developed models. In Europe, all smart meters are expected to have a granularity of 15 minutes [27], however, data sets generated for research purposes sometimes present smaller granularities.

The employed heating systems might also have an impact on electric load forecasting accuracy. For example, in France the house heating system is usually supplied through electricity while in other countries this is done through burning natural gas or coal. Furthermore, the presence of photovoltaic systems in houses might also affect the forecasting results, this is the case of the households from the Ausgrid data set.

When applying *Artificial Intelligence* techniques it is important that data sets are big enough so that the algorithm can extract all the useful information from these data sets. As it will be explained on a further chapter, small data sets might lead to overfitted models in which the error and not the real data set trends are modeled, this could lead to bad forecasting accuracies.

The first part of the work will focus on implementing LSTM RNNs and comparing the forecasting results with those obtained with ANNs in [43]. Therefore, for this part the same data sets as employed in [43] are used, these are:

- One household from the UCI data set
- Three households from the Smart data set

- Two household from the UK-DALE data set

The second part of this work will analyze the impact of data features on forecasting results. A larger data set will be analyzed to obtain conclusive results. The Ausgrid data set with 300 households will be employed to complete this part. Data sets with different characteristics (country, heating system, photovoltaic systems) are selected to obtain a general overview on the accuracy of the developed models.

2.4 Contribution

This work will focus on fulfilling three goals. The first one, will consist on developing a LSTM RNN model and compare it's forecasting results with the ANN model accuracy results obtained in [43]. To ensure a fair comparison, [43] used data sets will also be analyzed in this work.

The second objective will be to determine the main data set features affecting forecasting accuracy. Furthermore, motivated by the conclusions obtained in [24, 33] household´s behavioral patterns influence on forecasting accuracy will be analyzed. Results on a large number of household load data sets will be studied to extract significant conclusions.

The third objective of this work will be to determine the benefits of employing ANN or LSTM RNN depending on the data set characteristics. The impact of data set features on each forecasting model will be studied and conclusions on each model robustness will be extracted. These results will be compared to each model characteristics.

	Arora et al. [1]	**Taieb et al. [38]**	**Gerossier et al. [13]**
Methods	• Conditional kernel density estimation • Unconditional kernel density estimation • Holt-Winters-Taylor exponential smoothing	• Quantile regression • Unconditional quantiles • Additive models for location and scale parameters of normal distribution	• Quantile smoothing splines regression • Fallback model, use of surrounding smart-meters to overcome defective measures.
Dataset	CER	CER	SENSIBLE
Granularity	30 min	30 min	1 h
Forecast horizon	30 min - 7 days	30 min - 24 h	1 - 24 h
Input variables	• period of week • period of day • holiday / workday • lagged consumption value from the same time of the previous day	• period of week • period of day • time of year • holidays • lagged consumption values from the same time of two previous days • lagged consumption values of the previous six hours	• lagged consumption value from the same time of the previous day • median consumption of the hourly load during the previous week • local temperature prediction
Evaluation metrics	MAE, CRPS	CRPS	MAPE, NRMSE, CRPS

Table 2.1: Publications on probabilistic short term load forecasting on a household level

3 Method Fundamentals

This chapter is divided in two parts. The first part introduces the main concepts regarding probabilistic forecasts generation. Both the feedforward neural network and the LSTM recurrent neural network are explained. Furthermore, the evaluation metrics suitable to rank probabilistic forecasts are presented.

The second part of this work introduces the main concepts regarding time series data characterization, both traditional statistical features and clustering techniques are described. Furthermore, the metrics used to score smart meter data features and the correlation analysis employed to determine their impact on forecasting accuracy are presented.

3.1 Forecasting models

The goal of this thesis is to develop probabilistic forecasting methods by using Artificial Intelligence (AI) techniques. Short term load forecasting at a residential level is a very challenging task for two main reasons. The first reason is that individual loads have a high volatility, the second reason is that these loads largely depend on human behavioral patterns and these patterns are difficult to extract. During the last years many research studies [24, 28, 2] successfully used machine learning methods to improve forecasting accuracy.

Statistical techniques were traditionally used for residential load forecasting at an aggregated level, these models rely on hard-coded relationships between inputs and outputs. However, statistical techniques are not reliable with household electricity loads forecasting [33]. AI systems have the ability to acquire their own knowledge by extracting patterns from raw data. This capability is known as machine learning. The main advantage of using these techniques is that the relation between inputs, namely past loads, weather, calendar values etc. and the forecasts don´t have to be explicitly detailed but will be given by the model. By using Artificial Neural Networks (ANN) the input variables are defined but the parameter values are adapted during training.

A major source of difficulty in many real-world Artificial Intelligence (AI) applications is that many of the factors of variation influence every single piece of data we are able to observe. Extracting such a high-level abstract features from raw data can be very difficult. Deep learning solves this central problem by introducing features that are expressed in terms of other features. As it will be explained in the following section, this can be achieved by stacking multiple layers of neurons and relying on stochastic optimization to perform machine learning tasks.

The most successful NN model in the context of time series forecasting is the feedforward ANN, also known as *multilayer perceptron*. ANNs have proven to be universal function estimators without introducing further assumptions on the distribution [6]. However, at a household level, load consumption mainly depends on the behavioral patterns of its residents. As stated in the literature review, Long-Short-Term Memory (LSTM) Recurrent Neural Networks might be a suitable technique to effectively extract household patterns.

The fundamental difference between feedforward NN and LSTM RNN is the way information flows through the network. In feedforward NN the information flows straight through the network never passing the same node twice. On the other hand, recurrent networks take as an input not just the current input variables, but also variables that were previously processed. Recurrent networks have two sources of inputs, the present inputs and the recent past inputs. On the contrary feedforward networks have no notion of order in time, it only considers the current example it is exposed to.

3.1.1 Feedforward Neural Networks

Feedforward neural networks are the most typical example of a deep learning model, the goal of these models is to approximate some function $f*$. A feedforward network defines a mapping $y = f(x, w)$ and learns the value of the parameters w that result in the best function approximation. These models are called feedforward because information flows through the function being evaluated from x, through the intermediate computations used to define f and finally to output y. There are no feedback connections in which outputs of the model are fed back into itself. The fundamental building block of a feedforward network is the mathematical model of a neuron. As shown in figure 3.1, a neuron receives input from many other units and computes its own activation value.

There are three basic components inside one neuron. The first component is the connection links that multiplies each input variable by a weight w_j. The second component is the sum of the weighted inputs and the bias b, that is the input to the activation function. Finally, the output is computed by the activation function ϕ. The activation function decides the level of activation of each neuron. Mathematically this can be achieved by introducing a binary function, that is either the neuron is firing or not. However, in this case a small change in the sum component $\sum w_i x_i + b$ can cause the output to change it's state. Therefore, bounded sigmoid functions are usually chosen as activation functions. Sigmoid neurons are commonly employed for developing ANNs [15, 40].

Feedforward neural networks contain an input layer consisting of nodes that retain the input values, there are also successive layers of nodes that are neurons. The output values form one layer of neurons are the input values to the next layer. The last layer is called the output layer. All the layers placed between the input and the output layer are called the hidden layers.

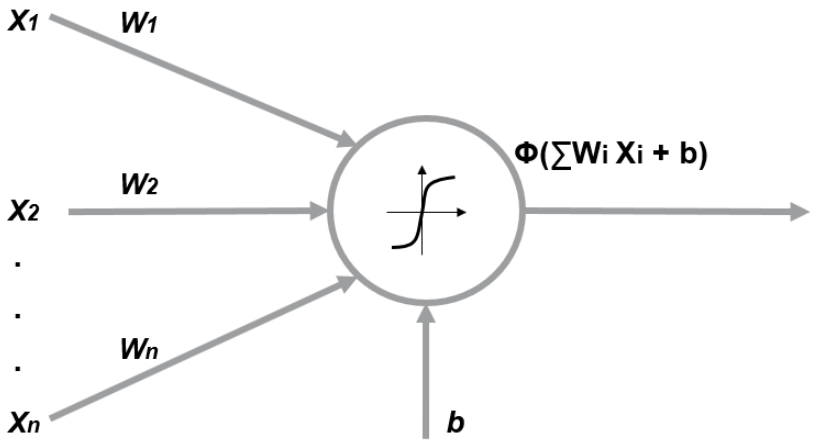


Figure 3.1: Representation of a single artificial neuron.

Figure 3.2 shows a multi-layer Neural Network. This network generates m outputs out of n input values. Networks with just one hidden layer are also called shallow Neural Networks while Deep Neural Networks present more than one layer of hidden neurons. As introduced in the previous section, introducing hidden layers enables extracting features from the data with a higher degree of abstraction. Unfortunately, there is no clear theory to guide the user on choosing the correct number of layers or nodes in each hidden layer. In this work, previously defined structures as well as a trial and error approaches are analyzed to determine the best network configurations.

Load forecasting is a supervised learning problem. Supervised learning is a type of machine learning problem in which the user has access to input and output variables and the developed algorithm learns the mapping from the input to the output. The goal is to predict the output variables by feeding an input to the model. On the contrary, an unsupervised learning problem just works with input data and its main goal is to find the underlying structure or distribution in the data. As it will be introduced in a further section, clustering algorithms belong to unsupervised machine learning problems.

Neural network optimization refers to the process of adapting all network's weights w_i and biases b_j to improve the predictions out of inputs values. The loss function $E(\hat{y}, y, w)$ quantifies the difference between the network output y and the training example $\hat{y}$ given a weight w. This error value is then used to adjust the weights of all the connections in the network using the backpropagation algorithm.

The backpropagation algorithm cycles through two different passes, a forward pass followed by a backward pass through the layers of the network. The algorithm alternates between these passes several times as it scans the training data. Typically, the training data must be scanned several times before the network *learns* to predict precise values.

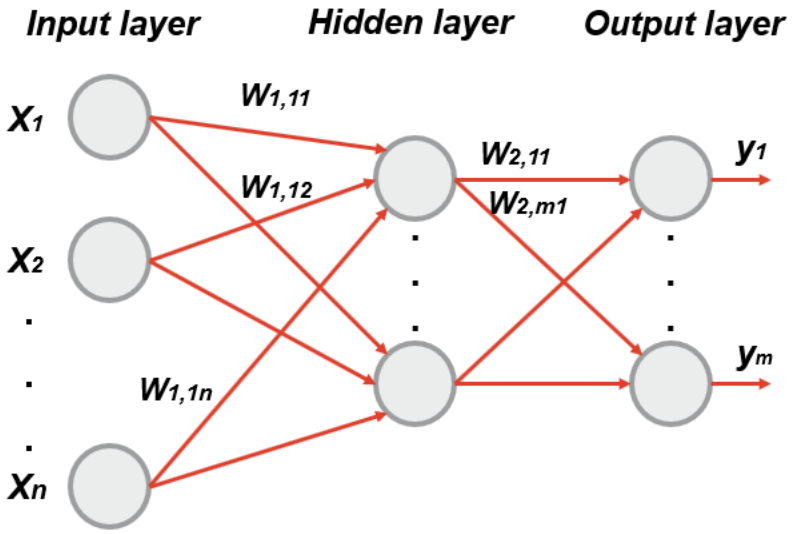


Figure 3.2: Representation of a Multilayer Neural Network with m outputs and n inputs.

The outputs of all the neurons in the network are computed in the forward pass. The algorithm starts with the first hidden layer using as input values the independent variables of a case from a training data set. The neuron outputs are computed for all neurons in the first hidden layer by performing the relevant sum and activation function evaluations. These outputs are the inputs for neurons in the second hidden layer. This process continues layer by layer until the output layer is reached. The resultant values constitute the model's dependent variable.

During the backward pass the propagation of errors and the adjustment of the neurons' weights are computed. This phase begins with the computation of errors at each neuron in the output layer. Once all errors are computed, the weights are updated proportionally to their gradient, which corresponds to their contribution to the loss function. The proportionality factor is called the learning rate α. The learning rate sets the step size at which the weights are changed, when the learning rate is set too small, the backpropagation algorithm will require many iterations to reach an optimum, while if the learning rate is set too big, the process of finding a local optimum might not converge. The following equation describes the update of $w_{ij}^{(l)}$, that is the weight between the *i-th* neuron in layer *l*-1 and the *j-th* neuron in layer *l*.

$$w_{ij,new}^{(l)} = -\alpha \frac{\partial E}{\partial w_{ij}^{(l)}} \tag{3.1}$$

Due to the complexity of the model and the large number of weights that are being adjusted as the network *learns*, there is no assurance that the backpropagation algorithm will find the optimum weights that minimize the error, the procedure can

get stuck in a local minimum. It is possible to speed up the algorithm by batching, that is updating the weights for several exemplars in a pass. A single scan of all cases in the training data is called an epoch. Most applications of feedforward networks require multiple epochs before errors are reasonably small. As it will be explained in the implementation section, the Adam optimization algorithm will be used to train the model.

As previously explained, Deep Neural Networks contain non-linear hidden layers and this makes them very expressive models that can learn very complicated relationships between their inputs and outputs. However, when handling with limited training data, these complicated relationships will model sampling error of the training data and not real data features. This leads to overfitting, resulting in a model with a high variance. A simple overfitting example is presented in 3.3.

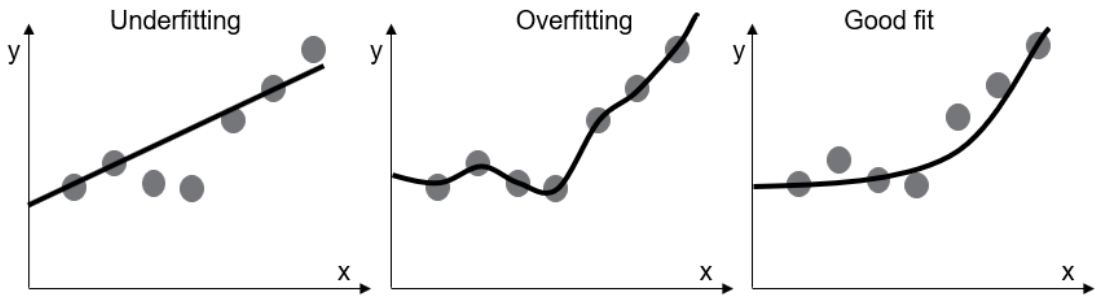


Figure 3.3: Underfitting, overfitting and good fit for a polynomial model $y = f(x)$

When developing Neural Networks, cross-validation can determine the overfitting behavior of the model. Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. In order to do so, an additional test data called cross-validation data must be used to generate unbiased test scores. In the case of handling data with high bias, the errors on the training and the cross-validation data sets will be high. On the case of overfitting the model, the training error will be considerably lower than the cross-validation error.

There are many commonly used methods to reduce overfitting. One option might be employing a larger data set, if this is not possible the user can always define a simpler neural network architecture. Another commonly used method consist on stopping the training as soon as performance on the validation set starts to get worse, the early stopping method is described in figure 3.4.

Furthermore, the dropout method reduces overfitting by randomly cutting connections between neurons during training. Dropout is a regularization technique for neural network models proposed in [34]. The technique consists on randomly selecting neurons that are ignored during the training. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight update is not performed on the backward pass. The effect of the

dropout technique is that the network becomes less sensitive to the specific weights of neurons. This results in a model less likely to overfit the training data.

As it will be introduced in the implementation section, in this thesis both the early stopping and the dropout method are used to prevent the overfitting of the model. Further information about dropout can be found in [34].

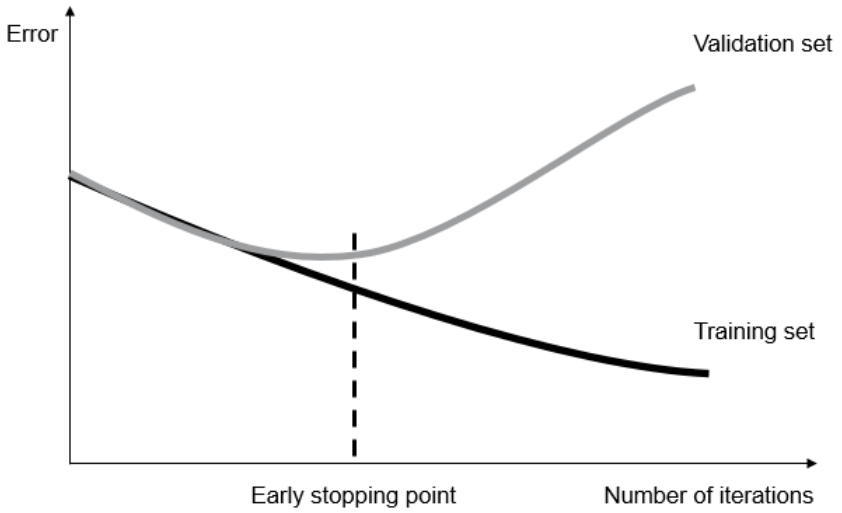


Figure 3.4: Iteration stopping to prevent overfitting

3.1.2 Long Short Term Memory Recurrent Neural Networks

Unlike feedforward neural networks, Recurrent Neural Networks (RNNs) are sequence-based models which can establish temporal correlations between previous information and the current circumstances. The structure of the network is similar to a feedforward NN, with the distinction that it allows a recurrent hidden state whose activation at each time is dependent on that previous cycle. The time recurrence is introduced by a memory state in the RNN hidden layers. For time series problems, the decision a RNN made at time step t-1 could affect the decision it will take at t. This characteristic of RNNs is ideal for residential load forecasting as residential's daily routines may be one of the most important factors to the energy consumption [24, 33].

RNNs are trained by backpropagation through time (BPTT) this is a modification of the backpropagation algorithm that works on sequences in time. However, as stated in [5, 18] learning long-rate dependencies with RNNs can be difficult due to gradient vanishing or gradient exploding problems. The exploding gradients problem refers to the large increase in the norm of the gradient during the training. Such events are caused by the rapid increase of the long-term components, which can grow exponentially more than short term ones. The explosion occurs by repeatedly multiplying gradients through the network layers that have values above 1.0.

Gradient vanishing refers to the fact that the norm of the gradient for long-term components decrease exponentially to zero making it impossible for the model to learn correlations between temporally distant events.

Long Short-Term Memory (LSTM) RNNs were introduced in the article [19] as a solution to the gradient vanishing and exploding problems. Unlike the traditional recurrent unit which overwrites it content each timestep, the LSTM unit is able to decide whether to keep the existing memory via introduced gates.

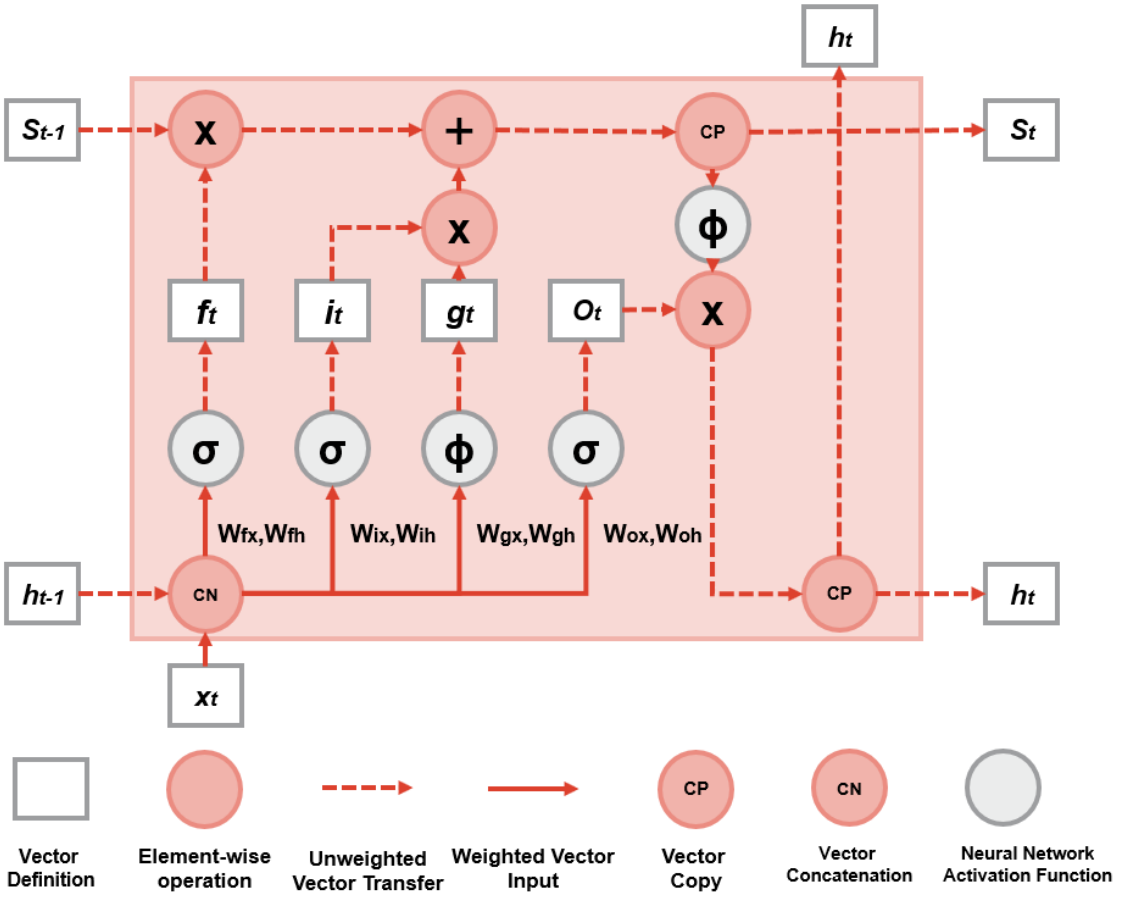


Figure 3.5: Structure of an LSTM block.

The LSTM memory block is represented in figure 3.5. $x_t \in \Re^k$ is a k dimensional vector $(x_1, x_2, ..., x_k)$ that represents the input values to the network at time step t. h_{t-1} is the output value from the selected LSTM block at time step $t-1$. s_{t-1} represents the memory cell state at time step $t-1$, the memory cell is the key feature from a LSTM block, it stores elements from past values that are updated, maintained or erased based on the outputs of the previous time step and the inputs of the present time step. In addition to the internal state, the LSTM structure

also defines input node g_t, input gate i_t, forget gate f_t and output gate o_t. The formulation of all nodes in an LSTM structure are given by:

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \tag{3.2}$$

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \tag{3.3}$$

$$g_t = \phi(W_{gx}x_t + W_{gh}h_{t-1} + b_g) \tag{3.4}$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \tag{3.5}$$

$$s_t = g_t \odot i_t + s_{t-1} \odot f_t \tag{3.6}$$

$$h_t = \phi(s_t) \odot o_t \tag{3.7}$$

$W_{gx}, W_{gh}, W_{ix}, W_{ih}, W_{fx}, W_{fh}, W_{ox}$ and W_{oh} are weight matrices for the corresponding inputs of the network activation function; $\odot$ stands for an element-wise multiplication; σ represents the sigmoid activation function, while ϕ represents the hyperbolic tangent function.

Each LSTM block presents three sigmoid functions acting as soft switches to decide which signals should pass the gates. The decisions for the forget gate f, the input gate i and the output gate o are all dependent on the current input x_t and the previous output h_{t-1}. The signal of the input gate controls what to preserve in the internal state, while the forget gate controls what to forget from the previous state s_{t-1}.

The LSTM have the ability to remove or add information to the cell state, this information flow is regulated by the three sigmoid functions that work as gates by returning values between 0, no information flow, and 1, full information flow. The forget gate f decides the information that has to be subtracted from the cell state. The input gate layer set the values that will be updated. The *tanh* layer creates a vector of new values g_t that could be added to the state. The old state s_{t-1} is multiplied by the forget gate and then added to the input and update gate.

Finally, a sigmoid function decides which part of the input variables and last output will impact the output. Then the cell state information is passed through an *tanh* function and the result is multiplied by the output of the sigmoid gate.

3.1.3 Probabilistic forecasting model

Probabilistic forecasts assign probabilities to future values. In this work the focus will be on developing forecasted future load values as probability density functions PDF.

The objective when developing probabilistic forecast is to maximize the reliability, sharpness, and resolution. Reliability refers to how close the predicted distribution is to the actual one. For example, if 50% of the observed values fall within 50% of the predicted density function.

Sharpness refers to how tightly the predicted distribution covers the actual one. If the maximum and minimum observed values are very close to the predicted maximum and minimum values, then the interval forecast is said to be sharp. Resolution refers to how much the predicted interval varies over time. High resolution refers to a high variance of forecasts. For further explanation refer to [37].

On [43] both mixture density networks MDN and softmax distribution networks SDN are employed to develop probabilistic forecasts. In the MDN method is represented as a linear combination of kernel functions, Gaussian kernels were used within this work. The results showed that both density estimation methods achieved similar performance, furthermore training the MDN was proved to be less robust than training the SDN. In this work only the SDN method is used.

Softmax distribution networks SDN is a method used to approximate probability distribution functions (PDFs). Each output neuron represents the mean probability density for a fraction of the output space, in our case the output space will be a set of power intervals. These fractions are normally referred to as bins. All outputs are normalized so that the sum of all probabilities equals one. As a result, each bin represents the probability that the future electric load lies within an specific power interval. This model is represented in 3.6.

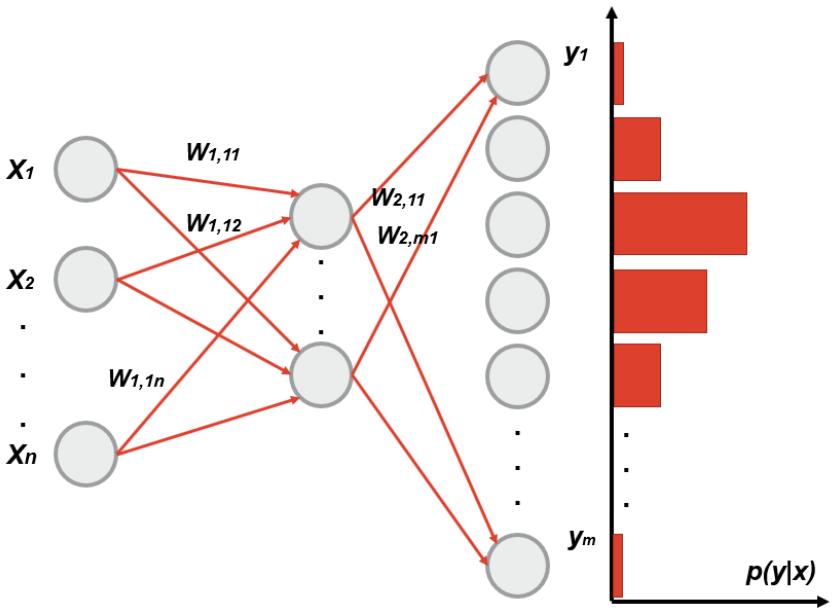


Figure 3.6: Representation of the softmax distribution network.

The negative log-likelihood algorithm will be used as the error function. In this case, the error from each output layer with i discrete bins is:

$$E(y, \hat{y}) = -lnp(y_{argmin_i|y_i - y|}x) \tag{3.8}$$

The network will be trained using the backpropagation algorithm defined in sections 3.1.1, the same algorithm will be used for feedforward neural networks and for LSTM recurrent neural networks.

3.1.4 Evaluation metrics

To obtain a good overview of the forecasting accuracy both deterministic and probabilistic evaluation metrics are employed. Deterministic evaluation metrics enable a model comparison with the *persistence method*. As stated before, the *persistence method* is a commonly used benchmark for deterministic STLF. On the other hand, it is required to evaluate the forecasting accuracy of the probabilistic STLF models developed and compare its results with previous studies [43]. In this work, the deterministic forecasting accuracy is determined by the mean absolute error MAE, while the continuous ranked probability score CRPS evaluates the probabilistic forecasting accuracy.

The mean absolute error MAE is a measure of difference between two continuous variables. This evaluation method compares the real household load y with the deterministic forecast value $\hat{y}$. The MAE returns the average error for N observations and forecast results.

$$MAE = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{N} \tag{3.9}$$

The continuous ranked probability score CRPS compares a full probability distribution function (PDF) with the observations, where both are represented as *cumulative functions*. The CRPS is defined as :

$$CRPS(F(\hat{y}), y) = \int_{-\infty}^{\infty} (\, F(z) - H(z - y))^{\,2} dz \tag{3.10}$$

In the previous function, $F(\hat{y})$ is the forecasted PDF, y is the observation. $H(z-y)$ is the Heaviside step function, which returns a one if the argument is positive and returns zero otherwise. The CRPS is expressed in the same unit as the observed variable, it generalizes the mean absolute error (MAE) for probabilistic inputs.

Finally, the negative log-likelihood NLL score is employed as the loss function in the backpropagation algorithm defined in section 3.1.1. This score is defined as:

$$S(p(y), \hat{y}) = -logp(\hat{y}) \tag{3.11}$$

The main advantage of using a logarithmic function is that the function becomes easier to operate, multiplications become sums. Furthermore, the logarithmic transformation converts small numbers into large negative values which a computer can operate better.

3.2 Time series data characterization

During the last decades there has been a substantial improvement of forecasting techniques. Despite the advances, even today it is not possible to determine what is the overall best model for short term load forecasting. In [37], T. Hong states that at an aggregated level "a universally best technique simply does not exist. It is the data and jurisdiction that determine what technique we should use".

The second part of this work aims to analyze which are the main time series data features affecting forecasting accuracy. The forecasting models used are the feedforward neural network and the LSTM recurrent neural network. This chapter presents the theoretical approach proposed to conduct the time series features impact study.

3.2.1 Relevant parameters for load profile characterization

As introduced in section 2.2, the smart-meter data features that are expected to have the biggest impact on the forecasting accuracy are:

- Average load value
- Time series cyclic behavior
- Load data dispersion
- Behavioral pattern complexity
- Behavioral pattern consistency

Furthermore, the data set seasonality is also considered. Seasonality refers to the presence of variations that occur at specific regular interval less than a year. Household electricity loads present a strong daily seasonality and a weaker week seasonality. Within this work daily load profiles are analyzed to extract household's behavioral patterns.

The average power load value is the first feature compared against forecasting accuracy. Further on, the impact of cyclic behavior is analyzed. Cyclic patterns exist when data exhibit rises or falls that are not of fixed period. A common example of cyclic pattern in household electricity consumption is a holiday period when the residents are not at home. Figure 3.7 shows the cyclic behavior of household load data.

Data set dispersion measures the extent to which a distribution is stretched or squeezed. The concept of pattern complexity refers to the number of well-defined behavioral patterns of a household's electricity consumption. Households with a high pattern complexity present many well-defined clusters depending on different day types. On the contrary, a household with low pattern complexity will have one well defined behavioral pattern. This concept was previously used in [24]. Daily load profiles are clustered to extract households' behavioral patterns.

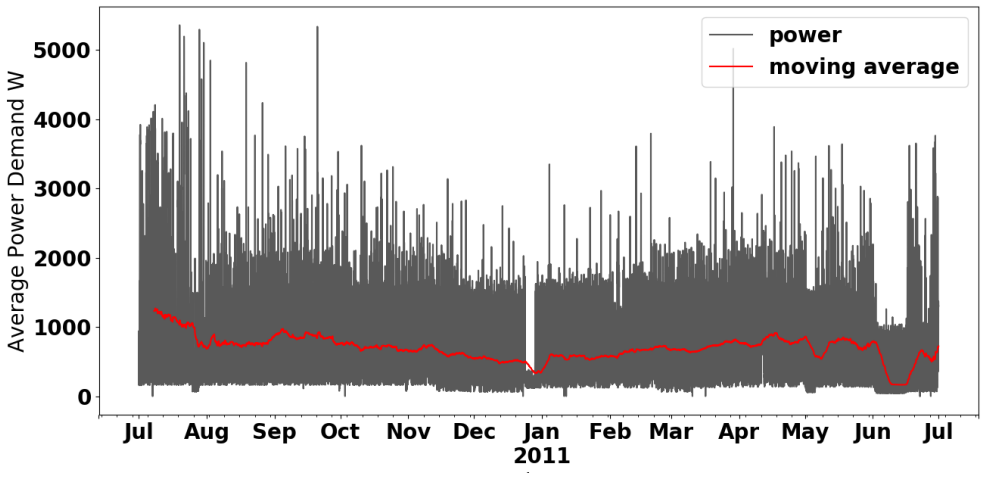


Figure 3.7: Original and moving average load values from Ausgrid household 1.

The last feature analyzed is the household's behavioral pattern consistency. As presented in [24] behavioral pattern consistency refers to the predictability of consumption data. Therefore, pattern consistency is the opposite of random consumption behavior.

3.2.2 Clustering techniques

In this work, clustering techniques are employed to capture temporal consumption patterns persisting in single-meter load profiles. The k-means clustering enables an understanding on the behavioral pattern complexity while Density Based Spatial Clustering of Application with Noise DBSCAN is used to measure the behavioral pattern consistency.

In the field of statistical learning, clustering is categorized as a form of unsupervised learning. This type of learning is formally described as a density estimation problem. As defined in [17] "The goal is to directly infer the properties of this probability density without the help of a supervisor or teacher providing correct answers or degree-of error for each observation". One main challenge of unsupervised learning is that it is often difficult to assess the results since there is no universally accepted mechanism for performing cross-validation results on an independent data set.

The goal of clustering is to find subgroups of clusters in a data set. Each subgroup contains elements that are similar to each other, thus it is important to correctly define the similarity between two objects. K-means clustering and Density Based Spatial Clustering of Application with Noise (DBSCAN) are two of the most cited household's load clustering algorithms in scientific literature [24, 42, 9].

3.2.3 K means clustering

As previously stated, the pattern complexity measure is obtained by performing a k-mean clustering. Daily load profiles are clustered to extract household's behavioral patterns. The pattern complexity measure employed is the number of daily profiles in the biggest k-mean cluster. If most daily profiles fall in the same cluster the pattern complexity is considered to be lower than if on the opposite all daily clusters have similar sizes.

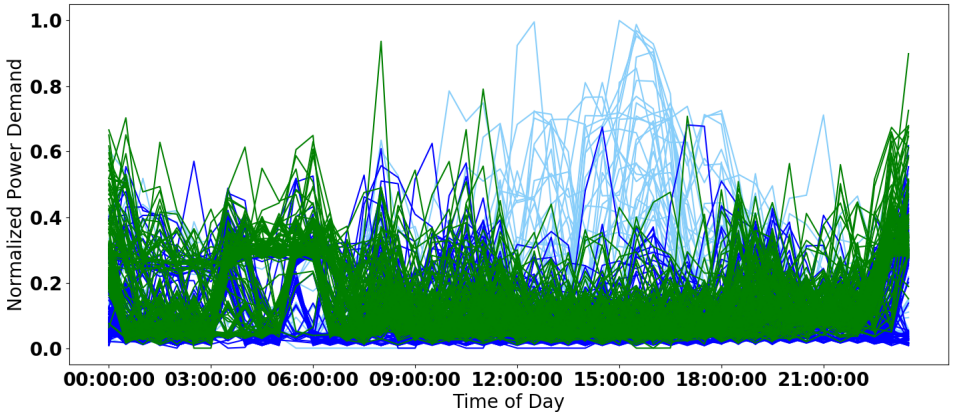


Figure 3.8: K means clustering of Ausgrid household 1 normalized data.

We begin by considering the problem of identifying groups, or clusters, of household daily load profiles. Supposing that a data set $(x_1, x_2, ..., x_N)$ consisting of N smart meter daily profiles is given. The objective will be to partition the data set into K clusters. The result of the algorithm will be a set of clusters, in which each cluster is a group of profiles whose inter-point distances are small compared with the distances to points outside of the cluster. We can formalized this concept by first introducing a vector μ_k, where $k = 1,...,K$, that is a prototype associated with the k^{th} cluster. μ_k represent the center of each cluster. The objective is then to find an assignment of data profiles to clusters, as well as a set of vectors μ_k, such that the sum of the squares of the distances of each data profile to its closest profile μ_k, is a minimum.

For each daily profile x_n, we introduce a corresponding set of binary indicator variables $r_{nk} \in 0, 1$, where $k = 1,...,K$ is the number of clusters set by the user. r_{nk} describes if daily profile x_n is assigned to cluster k, $r_{nk} = 1$, or not $r_{nk} = 0$. The objective function to minimize in the k-means clustering algorithm is the distortion measure presented in the following equation.

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2 \tag{3.12}$$

The distortion represents the sum of the squares of the distances of each daily profile to its assigned vector μ_k. The objective is to find values for the r_{nk} and

the μ_k so as to minimize the distortion J. This is usually done through an iterative procedure in which each iteration involves two successive steps corresponding to successive optimizations with respect to r_{nk} and μ_k. The K-means algorithm provides a simple solution to find the local optimum.

The optimal number of clusters K can be extracted by analyzing the relationship between the number of defined clusters K and the distortion measure of the clustered data set. The distortion will decrease as the number of clusters K increases. The *Elbow method* is graphical method used to estimate the optimal number of clusters. The main idea is to identify the value of K where the distortion begins to decrease more slowly, hence the *elbow criterion*. However, this *elbow* cannot always be unambiguously defined, this point will be discussed in the implementation section.

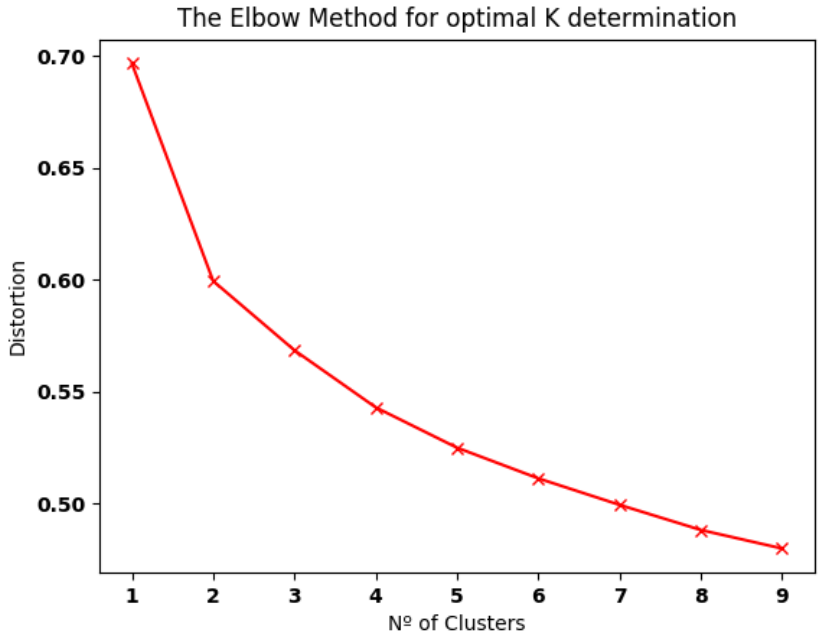


Figure 3.9: *Elbow method* UCI daily profiles data set.

Figure 3.9 depicts an example of employing the *Elbow method*. This graph shows that the most suitable number of clusters for this data set will be two. This is due to the fact that the dispersion decreases slower if more than two clusters are defined.

3.2.4 DBSCAN clustering

The number of daily consumption profile outliers accounts for the pattern randomness, which is the opposite feature from the pattern consistency. Time series outliers are profiles that do not follow the general historical pattern of regular variation seen in the data sequence. The Density Based Spatial Clustering of Application with

Noise (DBSCAN) is used to extract the number of outliers from daily consumption profiles.

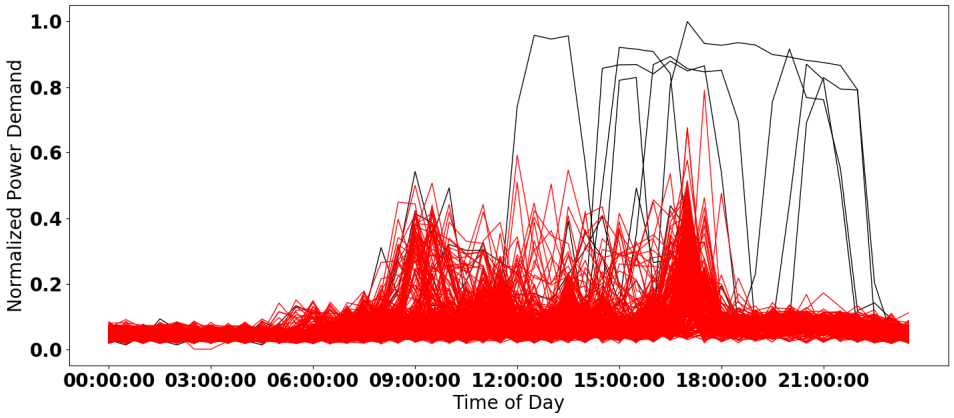


Figure 3.10: Daily consumption profiles with outliers.

DBSCAN is a density-based clustering technique which estimates similarities between daily profiles from a data set with respect to a predefined distance and partitions them into subsets known as clusters, so that the profiles in each cluster share some common patterns.

DBSCAN is designed to discover clusters and noise from a given set of profiles. This method classifies each profile either inside a cluster, in the edge of a cluster, or as neither a core profile nor a border profile, defining the profile as noise. It requires two main input parameters:

- *Eps*: maximum distance between two samples for them to be considered in the same neighborhood
- *MinProfiles*: number of samples in a neighborhood for a profile to be considered as a core profile

A further discussion about the procedure to choose these parameters is given in the implementation section. A profile is defined as a core profile if it has more than a specified minimum number of profiles required to form a cluster *MinProfiles* within an *Eps*-neighborhood. These are profiles that are in the interior of a cluster. A border profile has fewer than *MinProfiles* within *Eps*, but it is in the *Eps*-neighborhood of a core profile. A noise profile, or outlier, is any profile that is neither a core profile nor a border profile.

The DBSCAN algorithm proceeds as follows. First an arbitrary profile p is selected and all density-reachable profiles are retrieved. Second a decision rule is applied, if p is a core profile that contains at most *MinProfiles* a cluster is formed; otherwise, label p is an outlier. Third, a new unvisited profile is retrieved and processed leading to the discovery of further clusters of core profiles, this step is

repeated until all profiles have been labeled. Fourth, label any profile not belonging to a cluster as an outlier.

3.2.5 Metric definition

The metrics proposed in this thesis to describe the data features are:

- Average load value: Time series mean power value
- Time series cyclic behavior: Moving average standard deviation
- Load data dispersion: Median daily standard deviation
- Behavioral pattern complexity: Size of the main k-means cluster
- Behavioral pattern consistency: Number of outliers

A moving average (MA) is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. In this work the moving average method is used to extract the presence of cycles in the time series data. The selected method is the simple moving average (SMA) that returns the unweighted mean of the previous n data. In the following equation, the moving average (MA) at time t is computed, d_t stands for demand at time t.

$$\overline{MA_t} = \frac{d_t + d_{t-1} + ... + d_{t-(n-1)}}{n} \tag{3.13}$$

The selected time length depends on the type of movement of interest. When analyzing smart meter data, a commonly used window length n is one week. By selecting one week all the daily and weekly fluctuations are smoothed out. The metric that accounts for the cyclic behavior is the standard deviation of the moving average trend.

$$cycle = \sqrt{\frac{\sum (MA_i - \widehat{MA})^2}{N-1}} \tag{3.14}$$

The second metric of interest accounts for the load data deviation. The median daily standard deviation measures the dispersion of daily consumption data. The standard deviation of all daily profiles is computed and the median value of these deviations is extracted. The median and not the mean deviation is selected to prevent extreme values from affecting the result.

$$\widetilde{\sigma_{daily}} = median\left(\sqrt{\frac{\sum (d_i - \widehat{d})^2}{N-1}}\right) \tag{3.15}$$

The effect of pattern complexity on the forecasting accuracy is measured by the size of the biggest k-means cluster. K-means clustering is a popular method for

cluster analysis in data mining. This method partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

Pattern consistency is accounted for by the number of outliers within the daily load profiles. An outlier is an observation profile that is distant from other observations. In this work the DBSCAN clustering method is employed to determine the number of outliers in the data. This approach was previously conducted in [24].

The continuous ranked probability score (CRPS) is the metric used to account for the forecasting accuracy. CRPS results are compared with the data set feature metrics. Each data feature metric is compared with the CRPS values by means of a correlation analysis.

The Pearson correlation coefficient (ρ) is employed to measure the correlation between data features and accuracy results. This coefficient measures the linear correlation between two variables. It has a value between -1 and 1. A ρ of 1 means that there is a total positive linear correlation between variables, -1 means that there is a total negative linear correlation and 0 means that there is no linear correlation between variables. This value is calculated by:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{3.16}$$

Additionally, the p-value is computed. The p-value is defined as the probability, under the null hypothesis H, of obtaining a result equal or more extreme than what was actually observed. In this work it is used to indicate the probability of an uncorrelated system producing data sets that have a Pearson correlation at least as extreme as the one computed.

The threshold value for p, called significance level (α) will be set at 5%. If the p-value is less than α then the null hypothesis may be rejected. The p-value is therefore a tool for deciding whether to reject the null hypothesis.

4 Implementation

In this chapter the procedure to implement the forecasting models and to assess the data features impact on the forecasting accuracy is presented. The chapter begins with an introduction of all data sets used. The reasons for choosing each data set and their main characteristics are described. The development of forecasting models is introduced in the second section of this chapter, software and hardware choices are specified, as well as the workflow and the parameter selection processes. Finally, the data characterization and impact study implementation are described in the last section.

4.1 Data sets

Four publicly available smart meter data sets are used to develop this work. To enable a further model comparison all chosen data sets are publicly available. In fact, the selected data sets were developed for research purposes and used in previous research studies. The selected smart meters data sets are shown in table 5.1.

Name	**Granularity**	**Recording period**	**Location**	**Households**	**Ref.**
Smart*	1 min	3 months	MA, USA	3	[4]
UCI	1 min	47 months	France	1	[39]
UK-DALE	6 s	5 - 21 months	UK	3	[22]
Ausgrid	30 min	12 months	Australia	300	[27]

Table 4.1: Data sets

The first three data sets are employed to analyze the differences between the forecasting accuracy of the feedforward neural network model developed in [43] and LSTM neural network model developed in this work. To enable a fair comparison, the data sets used to complete this part of the work are the ones employed in [43].

A time series data characterization is conducted in the second part of this work. The objective is to analyze how different time series data features impact both models' forecasting accuracy. This study requires a data set with a large number of households, therefore the Ausgrid data set was chosen to complete this task.

4.1.1 Smart*

The Smart* project seeks to optimize energy consumption in homes, with specific attention to *smart homes* and the new opportunities made available by such homes. It is a project from the laboratory for Advanced Software Systems of the University of Massachusetts Amherst. The analyzed data set was released in the year 2013 and contains environmental data and aggregated electrical data for all households.

The first household, referred to as Household A is a two-story house without central air conditioning (A/C), the house heating system uses natural gas. The second household referred to as Household B has a similar size as Household A, in this case the house has a central A/C, this house also uses natural gas for heating. Finally, Household C has a much bigger size than Households A and B, it has solar panels and two micro wind turbines. The heating system also uses natural gas.

4.1.2 UCI

The UCI electric power consumption data set contains minute-based measurements of a single home located near Paris, France. The data set contains 47 months of smart meter aggregated active and reactive power consumption data. In this work loads from November 2009 until December 2010 are analyzed. There are two reasons to justify this decision. The first one is that the other data sets have shorter recording period, the second reason is that electric load data sets with time periods longer than one year might present trends that could worsen the forecasting accuracy. The household has an electric water-heater, an A/C system and multiple house appliances.

4.1.3 UK-DALE

The UK-DALE data set was made available by the Department of Computing of the Imperial College London. It consists on a high resolution (16 kHz recording) appliance level data from three houses. In this work the release of January 2015 is selected. All households are located in London.

Only the aggregated household loads are analyzed, the time length of the data sets are twenty months for Household 1, five months for Household 2 and five months for Household 3. All households are heated using natural gas and include two to four occupants.

4.1.4 Ausgrid

Ausgrid is an electricity infrastructure company owned by the government of New South Wales, Australia. They released solar home electricity data to help with analysis by research organizations, solar companies, governments and regulators.

The data has been collected from 300 randomly selected customers in Ausgrid's electricity network area. The data was gathered through meter reading processes

from 1 July 2010 to 30 June 2011. All customers have rooftop solar systems, their generation and load values are recorded every 30 minutes.

4.2 Forecasting models setup

The previous section introduced the data sets used to implement the forecasting models. This section describes the forecasting model development process. The first part introduces the input variables and the software and hardware choices selected to develop the forecasting models. The second part presents the developed modules and describes the relationship between the model development workflow and the code modules. Finally, the neural network model parameter selection is introduced in the last subsection.

4.2.1 Experiment definition

In this section the input variables are defined. As already explained, LSTM recurrent neural network model results are compared with feedforward neural network model results obtained in [43]. Hence, the same input variables as in [43] are used. Furthermore, this section presents the selected hardware and software employed for developing the forecasting models.

In [43] a grid search over the model hyperparameters was conducted. The objective was to find a suitable architecture of the neural network model and determine how the variable selection affects the forecasting accuracy. The input variables considered in this grid search study are presented in table 5.2.

Name	**Variables**
Lagged electrical load	$P_{t-1}, P_{t-2}, P_{t-3}, ..., P_{t-n}$
Time of the day	$hours \in [0, 23]$, $minutes \in [0, 59]$
Day of the week	$day \in [0, 6]$
Month of the year	$month \in [0, 11]$

Table 4.2: Input variables considered in [43]

The grid search results concluded that using a high number of lagged electrical loads can significantly improve the forecasting performance. Hence, within this work a length of 1440 minutes of lagged load values is considered to obtain the best possible forecasting results. On the other hand, as done in [43], time of the day, day of the week and month of the year will be considered as *calendar inputs*. As proven in [43], using *calendar inputs* is expected to have little impact on the forecasting accuracy, therefore in order to maintain the model as simple as possible *calendar inputs* will not be taken into consideration when forecasting future loads.

Furthermore, no weather variables are considered when developing the forecasting models. There are two main reasons for not considering weather variables. The first

one is the lack of weather-related data for most of the publicly available smart meter data sets. The second main reason is that weather variables are not expected to be a major factor of influence on short term electric load variations. As stated in [24, 43, 33] at a smart meter level, household behavioral patterns are expected to be the main factor of short-term load changes. [24, 43, 33] are other examples of STLF on a single household level without considering weather input variables.

The quantity that is forecasted within this work is the probability density function (PDF) of the total electrical energy consumption in the forecasting horizon $p(W_{el}|X)$, where W_{el} is the electrical energy consumption during the forecast horizon and X is the input vector, in this work the forecast horizon is set at 60 minutes. As explained in section 3.1.3, the PDF function is generated by a SoftMax distribution network (SDN). The output is in the form of $p(W_{el,1}, ..., W_{el,S}|X)$, that is the probability that future consumption reach the energy level $W_{el,i}$. Figure 4.1 illustrates the forecasting procedure.

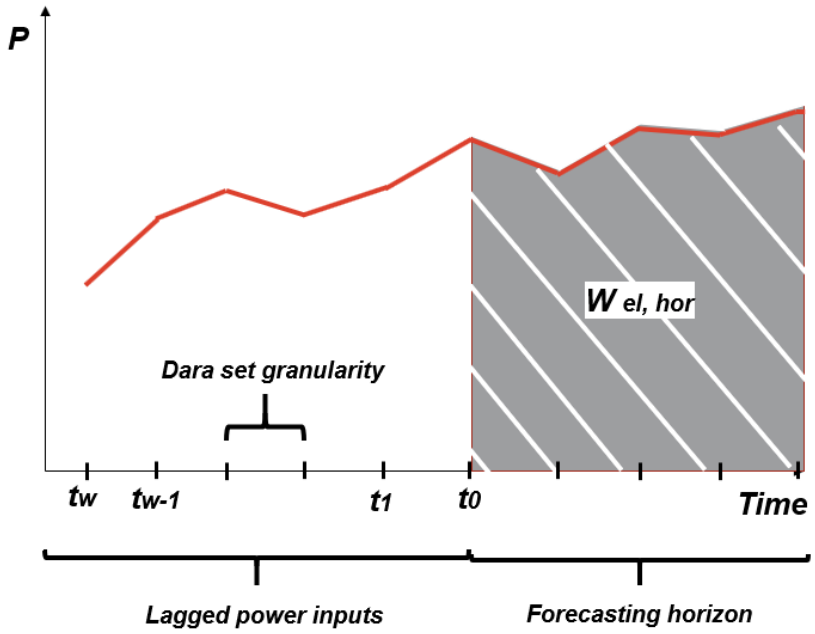


Figure 4.1: Energy consumption forecast

The software employed in this thesis is defined by the programming language and the libraries used to develop the forecasting models. Here Python is used as the programming language as it is the used language for implementing forecasting models at the E.ON Energy Research Center. It offers hardware-accelerated numerical computing and machine learning libraries as well as robust data handling packages that enables time-efficient prototyping and experimentation.

Training neural networks is a process that have high computational requirements. As seen in the previous section, the backpropagation algorithm search for optimal

weight values by performing a large number of matrix multiplications. Therefore, the use of GPUs is highly recommended when developing machine learning-based models. The GPU used was a NVIDIA Quadro K2200, when compared to an Intel Pentium this GPU showed a reduction of training times by factors around $20X$. The objective of this work is to compare the forecasting accuracy of two different neural networks models, therefore an extensive grid search was developed to achieve the most accurate models.

The numerical computation required to implement the neural networks is performed by an open source library called TensorFlow. TensorFlow has a flexible architecture that enables the user to deploy computation to one or more CPUs and GPUs in a destock or server. It was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purpose of conducting machine learning and deep neural networks research.

Keras is the selected high-level neural networks API running on top of TensorFlow. Keras is written in Python, it was developed with a focus on enabling fast experimentation. Among other things, using Keras enables easy and fast prototype development, it supports both feedforward and LSTM recurrent neural networks and it runs on CPUs and GPUs.

SciPy is an open source Python library used for scientific and technical computing [21]. The SciPy packages used in this work are: Numpy, used for mathematical functions, Matplotlib, used for 2D plotting, and Cluster, used for k-means clustering. Furthermore, the scikit-learn package is used for DBSCAN clustering. Scikit-learn is a data mining and data analysis library built on top of NumPy, SciPy, and Matplotlib.

4.2.2 Modules architecture

The forecasting models implemented within the scope of this work will be used by other researchers from the E.ON Energy Research Center. Therefore, the developed code should be easy and intuitive to implement, also a high modularity adds flexibility for further research studies. All the modules developed are compatible with the modules developed in [43]. This section describes all the implemented modules and presents the workflow process to generate and evaluate load forecasts, this process is presented in figure 4.2.

Each created module performs a different function. Once the smart meter data set has been downloaded, the *preprocessing.py* module prepares the data and converts it into a predefined format. In this work the predefined format consists on a csv file with two columns, the first column called *timestamp* records the time in the format $yyyy-mm-dd\ hh:mm:ss$. The second column named *power* records the power value measured in watts. At this step, the user defines the preprocessed data set granularity. Raw data sets have initial granularities ranging from 6 seconds to 1 minute. However, the scope of this work is analyzing granularities of 1, 5 and 30 minutes. The power output values are the mean power values for each time interval.

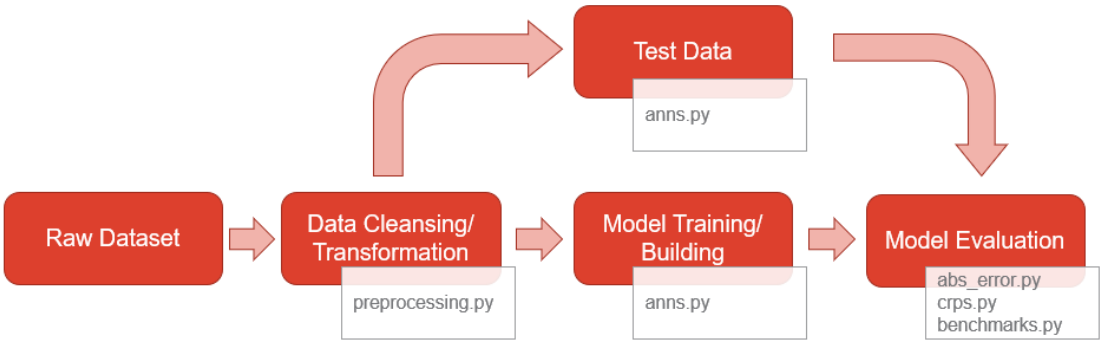


Figure 4.2: Neural networks model development workflow

To ensure an easy implementation on future research studies, all used directories are set in the *config.py* module. The idea is that the user will only have to set their directories in this module to use both forecasting models. The *utils.py* module generates the filename in which the preprocessed files are saved and creates a data frame with all preprocessed data content.

The *anns.py* module contains both the feedforward and the LSTM recurrent neural network forecasting models. This module includes the final data preparation, data split, model training and data forecasting. As introduced on a previous section, the data is split in training, validation and test data set, 80% of the data will be used for training, 10% for validation and 10% for testing. The model training is performed on the training set, the validation set prevents the model from overfitting. The trained model is then employed in the test data set to forecast load values.

Adam is the optimization algorithm used in this work [23]. Adam can be used instead of the classical stochastic gradient descent procedure to update network weights based on the training data. This method adapts the learning rates to make the optimization process more efficient. As mentioned on section 3.1.4, the negative log-likelihood is used as the loss function for the model optimization.

The final step is evaluating the forecasted probability distributions. These distributions are compared against the real load values by using the previously introduced metrics. All evaluation modules are placed inside the evaluation folder. In this work, the used metrics are the mean absolute error (MAE) values, developed in *abs_error.py* and the continuous ranked probability score (CRPS) developed in *crps.py*, the average CRPS value of all forecasted powers is used.

The evaluation procedure also compares the forecasted results with the persistence method results. This method is implemented in the *benchmarks.py* module, the module generates the output value from the selected data set. The user must introduce the granularity and the forecasting horizon. Once the persistence method output is generated, the MAE results are obtained to compare the forecasting accuracy of the persistence method against the accuracy of the neural network models.

A grid search is employed to search for the most accurate models. Different network architectures and dropout factors are included to search for the best model. *gridSearch.py* calls *anns.py*, *abs_error*, *crps.py* and *benchmarks.py* modules to train the forecasting models, forecast the test data set and evaluate its forecasting accuracy. Finally, all results obtained in the grid search are analyzed by the module *gridSearchScan.py*, this module returns the best forecasting models and plot the error results to analyze the hyperparameter impact on forecasting accuracy.

4.2.3 Model parameters

The work realized in [43] presented an analysis of the effects of the model hyperparameters on the forecasting accuracy. Although some interesting conclusion were extracted from this work, the results also enforced the fact that the optimal model parameters depend largely on the selected data set. In this work *calendar values* are not used, and the length of lagged inputs will be set at the highest value of 1440 minutes.

There are some model parameters that have to vary to obtain the best possible forecasting results. The first parameter of variation is the data set granularity. Data granularity describes the resolution of the data, in this work granularities of 1, 5 and 30 minutes are considered. Although using smaller granularities tend to improve the forecasting performance, it also largely increases the training time.

As stated in [15], the level of abstraction that a model can extract from the data increases with the complexity of the model. The number of stacked layers and the number of neurons per layer account for the network complexity. However, highly complex models tend to overfit the data, specially if the length of the data is not long enough. On the other hand, as the complexity of the model increases, so does the computational requirements and therefore the time to train the model. The selected number of layers implemented are 1 and 3 and the number of neurons per layer varies between 10, 40 and 100.

The dropout technique is implemented to prevent the model from overfitting. Nodes are selected randomly to be dropped-out given a probability, e.g. 20%. This is how dropout is implemented in Keras. In this work dropout values of 0%, 20% and 50% are implemented.

As introduced in the method fundamental's chapter, early stopping technique is also commonly used to prevent the model from overfitting. Early stopping is directly implemented by Keras at the training stage. The input values are the *min_delta*, that is the minimum variation at which the early stopping is triggered, and the *patience*, meaning the number of consecutive iterations reaching *min_delta* before the simulation stops. Within all models, the *min_delta* was set to 0.01. On the other side, the *patience* parameter should prevent the model from overfitting but at the same time should enable the model to gain sufficient complexity, *patience* values lie between 10 and 20.

The batch size limits the number of samples to be shown to the network before a weight update is performed. It optimizes the network's training by defining how

many samples to read at a time and keep in memory. The number of epochs is the number of times that the entire training data set is shown to the network during training. LSTM recurrent neural networks are sensitive to the batch size. A batch size of 200 and an epoch number of 1000 are defined to train the models.

4.3 Data characterization setup

This section describes the implementation of the data characterization impact study. The objective is to extract which smart meter data features have an impact on forecasting accuracy. A data set with three hundred households is selected to perform the impact study. This study is conducted by means of a correlation analysis to determine which features have an impact on forecasting accuracy.

4.3.1 Modules architecture

As mentioned on the previous section, the idea of this work is to develop modules that are intuitive and easy to use, it is also necessary to implement a modular approach that ensures flexibility for further research work. The implementation of the data characterization follows four differentiated steps:

- Data set preprocessing
- Model training and forecasting execution
- Time series data analysis
- Correlation study

The data set preprocessing follows a similar approach as the one presented in the previous section. However, in this case the data set contains a large number of households. A preprocessed data set for each household is created. In order to make the code more efficient, the user has the possibility of selecting the households that are preprocessed. Also, unlike the data sets analyzed in [43], Ausgrid data sets have a granularity of 30 minutes, therefore lower granularities cannot be extracted. This process is implemented via the module *preprocessing.py*.

The data characterization task is performed both with normalized and with raw smart meter data. The module *normalizing.py* normalizes the power values by dividing all values by its peak value, a factor of 1000 then multiplies all normalized values to avoid calculating small numbers.

Data set training and forecasting is executed by the *gridSearch.py* module. The same approach as described in the previous section is adopted with the difference that all hyperparameters are maintained constant. Although parameter tuning is essential to obtain the best forecasting model, tuning 300 models for each household is very time-consuming for this work. Some rules of thumb for parameter selection are adopted. From the grid search results from the seven data sets analyzed in [43]

it has been concluded that neural networks with a sufficient degree of complexity tend to perform good most data sets, here a neural network architecture of three hidden layers with 40 neurons per layer is chosen. Furthermore, a dropout value of 0.2 effectively prevents overfitting and enables a good model training. A similar approach was conducted in [24].

Once that forecasting models are defined for all households, the main time series data features are extracted. *dataCharacterization.py* extracts the following data features:

- Average power load value
- Time series cyclic behavior
- Intraday data deviation
- Behavioral pattern complexity
- Behavioral pattern consistency

Section 3.2.5 describes the metrics used to extract these features. The impact study is conducted by means of a correlation analysis. The existing correlations between the model accuracy and the different data features are analyzed in the *impactStudy.py* module. The model accuracy is measured by the previously obtained CRPS value, which is the mean CRPS from all forecasted values.

The Pearson correlation coefficient (ρ) is employed to measure the correlation between data features and accuracy results. This coefficient measures the linear correlation between two variables. It has a value between -1 and 1. A ρ of 1 means that there is a total positive linear correlation between variables, -1 means that there is a total negative linear correlation and 0 means that there is no linear correlation between variables. Additionally, the p-value is computed for deciding whether to reject the null hypothesis. The data characterization process is described in figure 4.3.

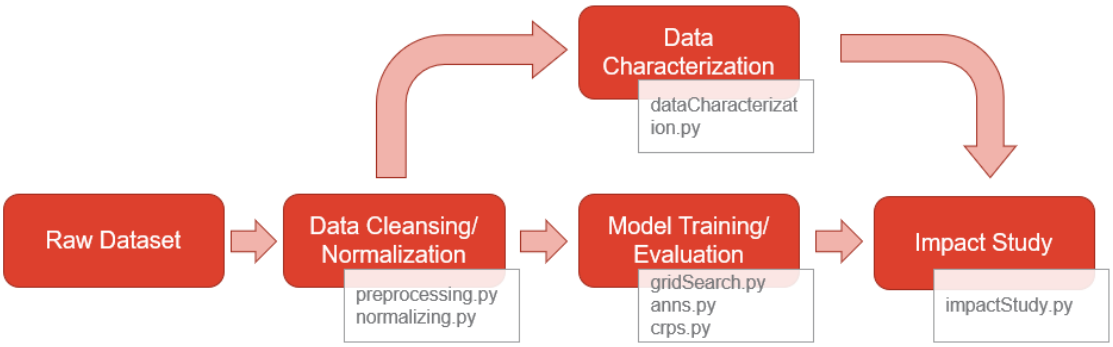


Figure 4.3: Data characterization and impact study workflow

4.3.2 Clustering

Cluster analysis or clustering is the task of grouping a set of objects in a way that objects in the same group, called cluster, are more similar to each other than to those in other clusters. In this work the daily electricity load profiles are clustered to extract the behavioral patterns within the smart-meters datasets.

The k-means clustering is implemented with the SciPy library. The k-means algorithm tries to minimize distortion, which is defined as the sum of the squared distances between each observation vector and its centroid.

As explained in section 3.2.3, the initial objective was to extract the optimal number of clusters K by using the *Elbow method* and set K as a measure of the behavioral pattern complexity. However, results have shown that $K = 2$ for most of the data sets which means that this measure is not representative of the pattern complexity for each household.

The *kmeans* function returns two outputs, the centroids profile and the daily profiles assigned to each cluster. The selected inputs are the daily load profiles, the predefined cluster number k and the number of iterations. The number of cluster k is set at three to search for different day types within the data. An expected result might be differentiating between workdays, weekends and holidays.

After the k-means clustering is performed, the number of profiles in the main k-means cluster is extracted. This metric extracts the complexity of the behavioral patterns. The idea underlying this metric is that the forecasting accuracy improves if most daily load profiles follow one specific behavioral pattern.

Furthermore, DBSCAN clustering is implemented via the sklearn library [30]. This method groups a set of objects that are closely packed together, marking as outliers objects that lie in low-density regions. The required inputs for employing the DBSCAN algorithm are:

- Eps distance: minimum distance between two samples for them to be considered as in the same neighborhood
- Minimum number of samples: number of samples in a neighborhood for an object to be considered as a core object
- Data set

The best DBSCAN clustering performance is obtained when the Eps is set at the default value. The minimum number of samples in a neighborhood to be considered as a core object is set at 4 samples, clusters consisting on less than 4 days are neglected for being not significant. The data set's daily profiles are introduced as the input data. DBSCAN clustering is then performed to extract the number of outliers present in the data set. Outliers are daily load profiles that cannot be assigned to any cluster. The number of outliers determine the extent to which household's electricity load patterns are random.

5 Assessment

In this chapter the previously introduced forecasting methods are evaluated on the data sets selected in [43] and on the 300-household data from the Ausgrid data set. Furthermore, the results from the impact study of different data set features on the forecasting accuracy are presented.

5.1 Forecasting models comparison

This section presents the forecasting results comparison between feedforward and LSTM neural network models. Furthermore, both forecasting methods are compared with the persistence method to assess their forecasting performance.

First, a comparison of both models on the data sets analyzed in [43] is presented, the second subsection compares both forecasting methods on the 300-household smart meter data from the Ausgrid data set.

5.1.1 Result comparison with previous studies

This section describes the result comparison between the feedforward neural network model developed in [43] and the LSTM recurrent neural network developed in this work. Seven smart meter data sets with different characteristics are analyzed throughout this section.

A grid search is conducted to search for the best forecasting models, table 5.1 shows the hyperparameters chosen for all data sets. Different data set granularities, dropout factors and network architectures are implemented.

Parameter name	**Value**
Dataset granularity [min]	1, 5, 30
Forecast Horizon [min]	60
Length of lagged input [min]	1440
Dropout	0, 0.2, 0.5
Number of hidden layers	1, 3
Number of hidden neurons	10, 40, 100

Table 5.1: Grid search hyperparameter selection

The data set is chronologically split in training, validation and testing data sets. The model is trained on the first 80% of the data, cross-validated on the following 10% of the data and the forecasting model is tested on the final 10% of the data.

The grid search results obtained for the seven smart meter data sets are shown in table 5.2. This table shows the forecasting error of the feedforward, the LSTM neural network and the persistence method for all data sets.

In order to enable a comparison with persistence method results it is necessary to calculate the mean absolute error (MAE) from the feedforward and the LSTM neural network models. A probabilistic forecast can be transformed into a deterministic one by performing a weighted sum of all power interval values multiplied by their correspondent probability value. Once the equivalent deterministic forecast is obtained the MAE can be calculated by the procedure described in section 3.1.4.

The results show that both models perform considerably better than the persistence method. According to the MAE results, when compared to the persistence method, the feedforward neural network model improves the forecasting accuracy on average a 53.00% while the LSTM recurrent neural network model improves the MAE results on a 48.74%.

Data set	**Feedforward NN**		**LSTM NN**		**Persis**
	CRPS[W]	MAE[W]	CRPS[W]	MAE[W]	MAE[W]
Smart* A	256	341	224	335	812
Smart* B	330	422	344	454	644
Smart* C	405	449	340	563	1018
UCI	235	323	233	323	616
UK-DALE 1	97	124	94	126	187
UK-DALE 2	69	88	70	104	161
UK-DALE 3	117	158	122	173	616

Table 5.2: Overview over grid search results

On four out of the seven data sets the LSTM recurrent neural network model had better CRPS error results than the feedforward model. On average the LSTM model error was 5.43% lower than the forecasting error from the feedforward model.

On data sets with small CRPS errors both forecasting models perform very similarly. However, when the forecasting errors increase, the differences between both models are more important. Further analysis on the impact of smart meter data on both model's accuracies is presented in the following section.

The LSTM recurrent neural network model have higher computational requirements that the feedforward neural network model. Consequently, training the LSTM model takes considerably more time than training the feedforward model. On average training a LSTM neural network takes $20X$ times longer than training a feedforward neural network.

Table 5.3 presents the optimal hyperparameter values obtained for each data set. From the network architecture values it can be concluded that the forecasting net-

works need a high degree of complexity to perform good forecasts. Furthermore, introducing dropout normally improves the forecasting behavior. Although the optimal granularity levels differ between data sets, it is observed that as obtained in [43] smaller granularities tend to improve the forecasting accuracy. However, smaller granularities require longer training periods.

Data set	**Feedforward NN**			**LSTM NN**		
	Gran.	Dropout	Arch.	Gran.	Dropout	Arch.
Smart* A	5	0.2	[100,100,100]	5	0.2	[40,40,40]
Smart* B	1	0.2	[100,100,100]	1	0.2	[100,100,100]
Smart* C	1	0.2	[100,100,100]	1	0.2	[100,100,100]
UCI	1	0.2	[40,40,40]	5	0.2	[40,40,40]
UK-DALE 1	30	0.2	[100,100,100]	30	0.5	[100,100,100]
UK-DALE 2	1	0.2	[100,100,100]	1	0.5	[100,100,100]
UK-DALE 3	1	0.2	[100,100,100]	5	0.2	[40,40,40]

Table 5.3: Optimal parameter values obtained from grid search

5.1.2 Ausgrid data set model comparison

As introduced on the previous chapter, both forecasting models are used to forecast all three hundred household smart meter data from the Ausgrid data set. Given the limited time to develop this work, the neural network parameters are kept constant for all households, furthermore the feedforward and the LSTM neural network models employ the same parameters. The parameter selection process is described in section 4.3.1. The selected parameters are shown in table 5.4.

Parameter name	**Value**
Dataset granularity [min]	30
Forecast Horizon [min]	60
Length of lagged input [min]	1440
Dropout	0.2
Number of hidden layers	3
Number of hidden neurons	40

Table 5.4: Parameter values employed in the Ausgrid data set

5.5 shows the average error results from the 300 households from Ausgrid data set. The average MAE from the feedforward neural network model is 23.58% lower than the average error result from the persistence method. In the case of the LSTM neural network model, the MAE error is 27.07% lower than the MAE results obtained from the persistence method. The MAE values show that both forecasting models outperformed the benchmark method.

Data set	Feedforward NN		LSTM NN		Persis
	CRPS[W]	MAE[W]	CRPS[W]	MAE[W]	MAE[W]
Ausgrid	252	350	239	334	458

Table 5.5: Average error results of the 300 households from the Ausgrid data set

The CRPS error results from the LSTM neural network model are on average 5.16% lower than the feedforward neural network model values. Furthermore, the LSTM neural network model outperformed the feedforward model on 73% of the cases, 218 households presented better LSTM forecasting results.

There are many factors that affect the forecasting accuracy. It has been observed that the worst forecasting performance occur when the electric load behavior changes drastically in the testing data time period. This causes that the previously learned relationships during the training data are no longer representatives in the test data set. Figure 5.1 shows an example of a smart meter data set with high forecasting errors. The vertical lines show the division between the training, validation and test data sets. As it can be observed, the load values follow a different trend during the test data causing a bad forecasting performance, the feedforward neural network model had a CRPS error of $1473W$ while the LSTM neural network model had a CRPS error of $770W$. In this case, the LSTM neural network model seems to adapt better to the trend change than the feedforward neural network model.

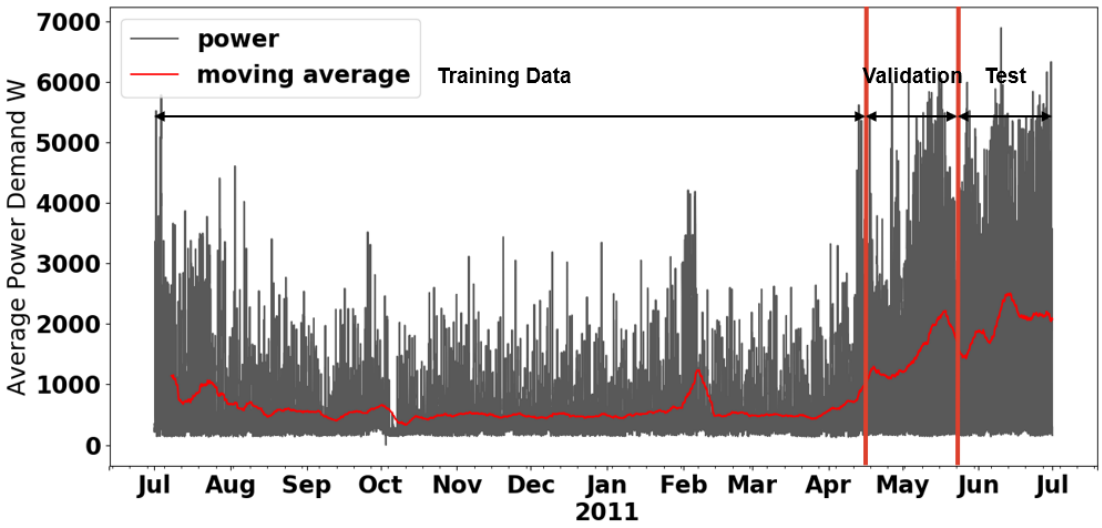


Figure 5.1: Ausgrid's household 96 load data

Furthermore, the benefits of using LSTM neural networks are easier to understand if a single household data is analyzed. Figure 5.2 shows the smart meter data from a household in which the LSTM neural network model CRPS error is 54% lower than the feedforward neural network CRPS error. A possible explanation for this difference is that the trends occurring in the test data time period are similar to

the ones that appear at the beginning of the training data set. The LSTM neural network model has a memory cell that might have been able to *remember* these past trends while the feedforward neural network is not able to store long time dependencies in data trends. The following section analyzes the data set features impact on the forecasting accuracy.

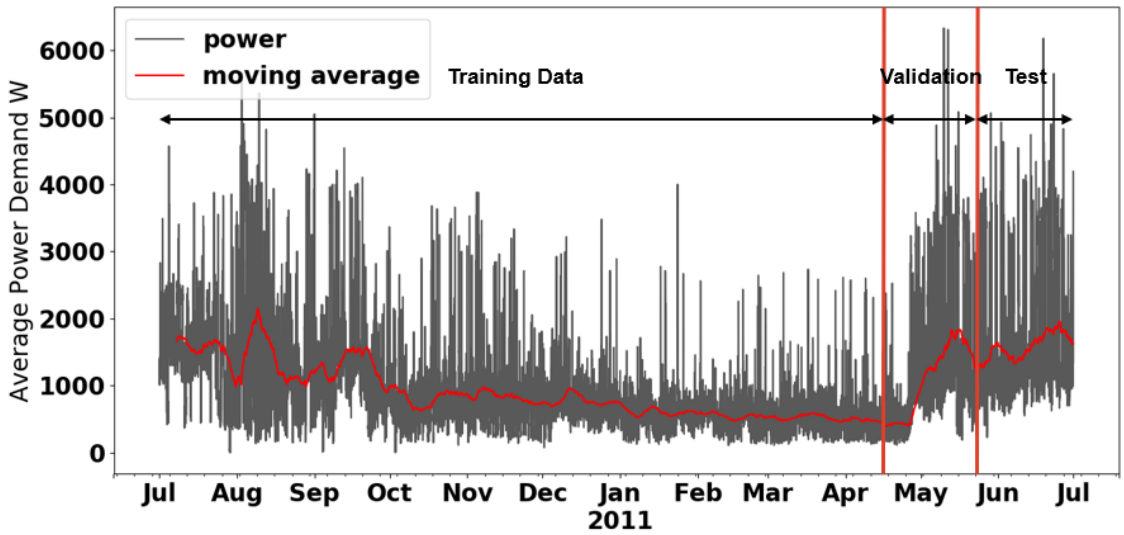


Figure 5.2: Ausgrid's household 192 load data

5.2 Data set feature impact on forecasting accuracy results

This section presents an impact study to extract which data set features have an impact on forecasting accuracy. The impact study is conducted by means of a correlation analysis. Each feature is compared with the forecasting error to determine possible relationships between these variables. Smart meter data from 300 households is analyzed to obtain reliable results.

First, traditional statistical measures are analyzed. Traditional measures include average load value, cyclic behavior and load standard deviation. On the other hand, clustering techniques are employed to extract the data set behavioral pattern complexity and behavioral pattern consistency.

5.2.1 Average electric power consumption impact on forecasting accuracy results

Due to its fluctuating nature it is usually difficult to forecast smart meter data. It might be expected that higher power consumption level lead to bigger power fluctuations and therefore to worse forecasting accuracies. In this section the average

electric power consumption from each smart meter data set is compared against the CRPS error results from the forecasting models. Figure 5.3 shows a scatter graph in which the average load power form each smart meter data is compared against its CRPS forecasting error.

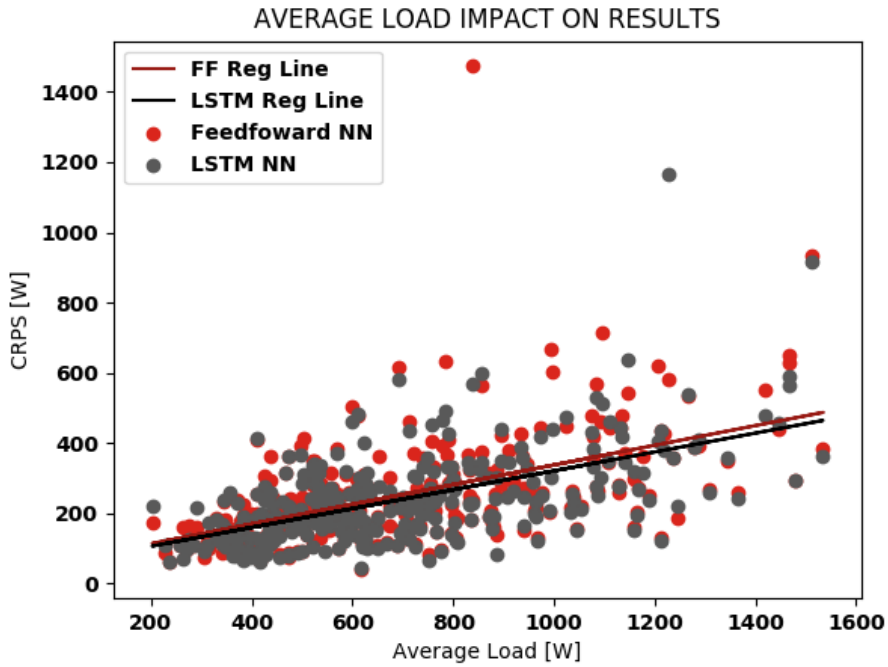


Figure 5.3: Average load impact results on the forecasting error of Ausgrid's household data

Each smart meter data set is represented with two points with the same power values but different CRPS error values. These points correspond with the feedforward neural network and the LSTM neural network model results for each household. There are 600 points represented on each graph.

Furthermore, the linear regression lines show the linear relationship of both model's error results with the average load value from each household. From the graphical results a positive linear relationship between the average load value and the forecasting error is expected for both forecasting models.

The correlation between variables is measured by the Pearson's correlation coefficient. The Pearson's correlation coefficient is used to measure the linear correlation between variables, furthermore the p-value is also extracted. As explained in section 3.2.5 the p-value measures the probability of a random point distribution obtaining a more extreme correlation coefficient than the one resulted from the given distribution.

The Pearson's correlation coefficient between the average load value and the feedforward neural network model CRPS error is 0.534, the p-value is $1 \cdot 10^{-23}$. For

the LSTM neural network model, the Pearson's coefficient is 0.527 and the p-value is $1 \cdot 10^{-28}$. Both p-values are lower than the significance value α set at 5%. This means that most probably there is a correlation between the analyzed feature and the forecasting error. Given these results, it can be affirmed that in the Ausgrid smart meter data there is a positive correlation between the average load levels and the forecasting error.

5.2.2 Cycle impact on forecasting accuracy results

Cycle refers to the presence of rises and falls in the data that are not fixed in time. As explained in section 3.2.5 data cyclic behavior is extracted by the standard deviation of the smart meter load moving average. It is expected that a fluctuating electric load moving average represent trend variations and therefore negatively affects the forecasting accuracy.

Figure 5.4 shows the impact of the standard deviation of the moving average on the forecasting error, measured by the CRPS error. The regression lines appear to show a positive relationship between the moving average standard deviation and the forecasting errors. Moreover, the Pearson correlation coefficients are 0.572 and 0.554 for both the feedforward and the LSTM neural network models. The p-values for these models are $1 \cdot 10^{-27}$ and $1 \cdot 10^{-25}$. The p-values are lower than the significance value α set at 5%. Given these results it can be assumed that for the analyzed data there is a positive linear correlation between the data cyclic behavior and the forecasting errors.

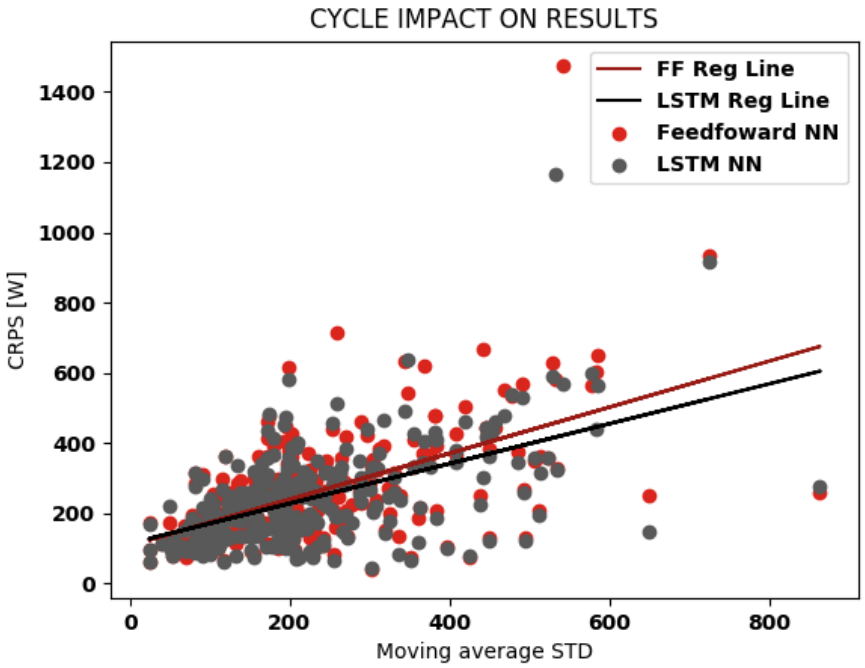


Figure 5.4: Cycle impact results on the forecasting error of Ausgrid's household data

5.2.3 Data standard deviation impact on forecasting accuracy results

In this work, the median daily standard deviation measures the smart meter data dispersion. A higher fluctuation is expected to have a negative impact on the forecasting accuracy. The distribution samples from the smart meter standard deviation impact on the CRPS error are shown in figure 5.5.

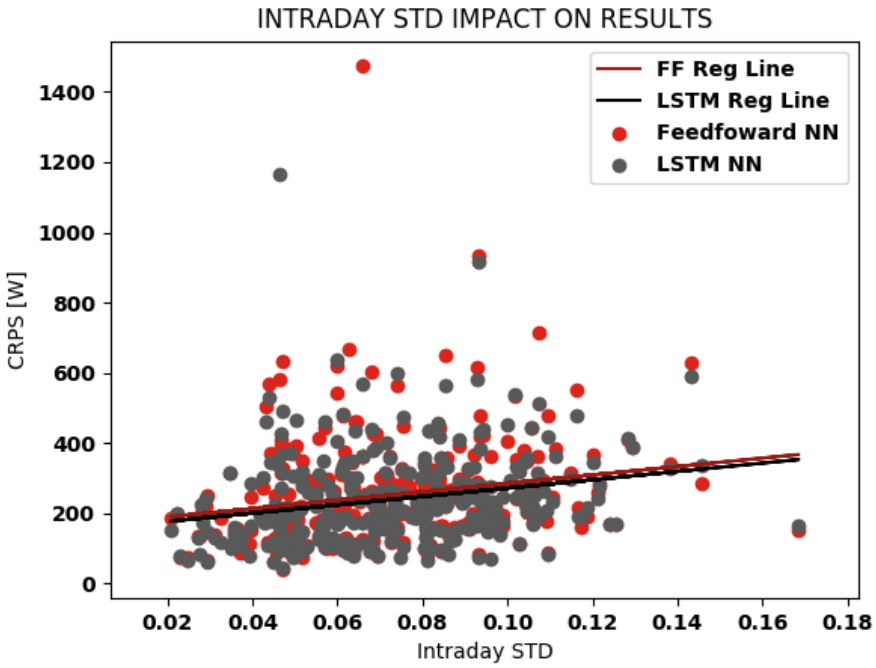


Figure 5.5: Daily standard deviation impact results on the forecasting error of Ausgrid's household data

Given the graphical results, the daily standard deviation seems to have a lower impact on the forecasting accuracy than the aforementioned cyclic behavior or average power consumption. The Pearson's correlation coefficients are respectively 0.201 and 0.223 for the feedforward and the LSTM neural network models. The p-values are higher than in previous cases, $4 \cdot 10^{-4}$ in the case of the feedforward neural network model and $9 \cdot 10^{-5}$ for the LSTM model. This means that there is a probability of 0.04% and 0.009% that a random point distribution have higher Pearson's correlation coefficients than the ones obtained. As these values are bellow the significance value α, it can be assumed that for the analyzed households there is a positive correlation between the median daily standard deviation and the forecasting error.

5.2.4 Behavioral pattern complexity impact on forecasting accuracy results

The behavioral pattern complexity refers to the number and frequency of well defined clusters. A high behavioral pattern complexity corresponds to multiple well defined daily profile patterns with equal number of daily profiles per pattern cluster. In low pattern complexity smart meter data most daily load profiles are assigned to the same pattern cluster. In this work the behavioral pattern complexity is extracted by conducting a k-mean clustering. The number of predefined clusters is set to three.

The defined metric consists on the main clusters' daily profile number. If most daily profiles fall in the same cluster the behavioral pattern complexity is expected to be lower, a lower pattern complexity is expected to result in better forecasting accuracies.

The graphical results are shown in figure 5.6. From the regression lines no linear correlation between the behavioral pattern complexity and the forecasting error is expected. Moreover, the Pearson's correlation coefficients are low, 0.044 in the case of the feedforward neural network model and 0.054 for the LSTM neural network model. Furthermore, these results seem to be non significant, the p-values were 0.45 and 0.35 meaning that there is a probability of 45% and a 35% respectively that 300 random values have a more extreme Pearson's correlation coefficient than the one extracted. Therefore, in can be concluded that in the Ausgrid data set the behavioral pattern complexity metric employed does not have any linear relationship with the forecasting accuracy.

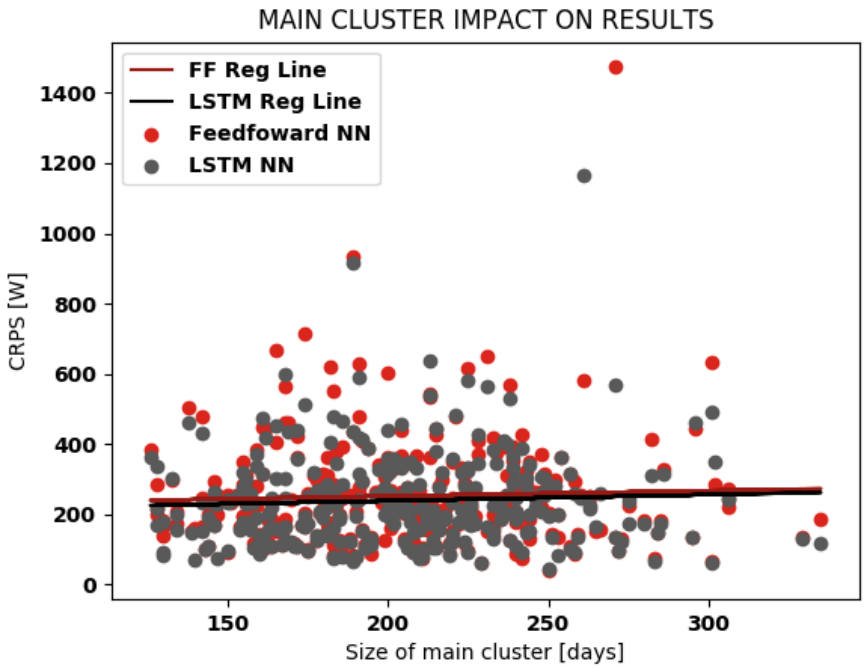


Figure 5.6: Behavioral pattern complexity impact results on the forecasting error of Ausgrid's household data

5.2.5 Behavioral pattern consistency impact on forecasting accuracy results

This feature describes the extent to which household smart meter data follow specific patterns throughout the whole data set. The behavioral pattern consistency is extracted by measuring the random behavior in daily load profiles, the random

behavior is an opposite concept to the pattern consistency it may therefore be used to extract this feature. As a metric, the number of outliers in household's daily load profiles is extracted, outliers are daily profiles that do not belong to any existing cluster. The outliers number are obtained by the DBSCAN clustering method, this method is explained in section 3.2.4.

The linear regression lines shown in figure 5.7 indicate a positive linear correlation between the number of outliers detected in each smart meter data set and the forecasting error. The Pearson correlation coefficients are 0.371 for the feedforward neural network model and 0.369 for the LSTM neural network model. The p-values assigned to this correlation coefficients are $3 \cdot 10^{-11}$ and $4 \cdot 10^{-11}$, respectively. The obtained results confirm the hypothesis for this data set, there is a positive linear correlation between the presence of outliers in smart meter data sets and the forecasting error.

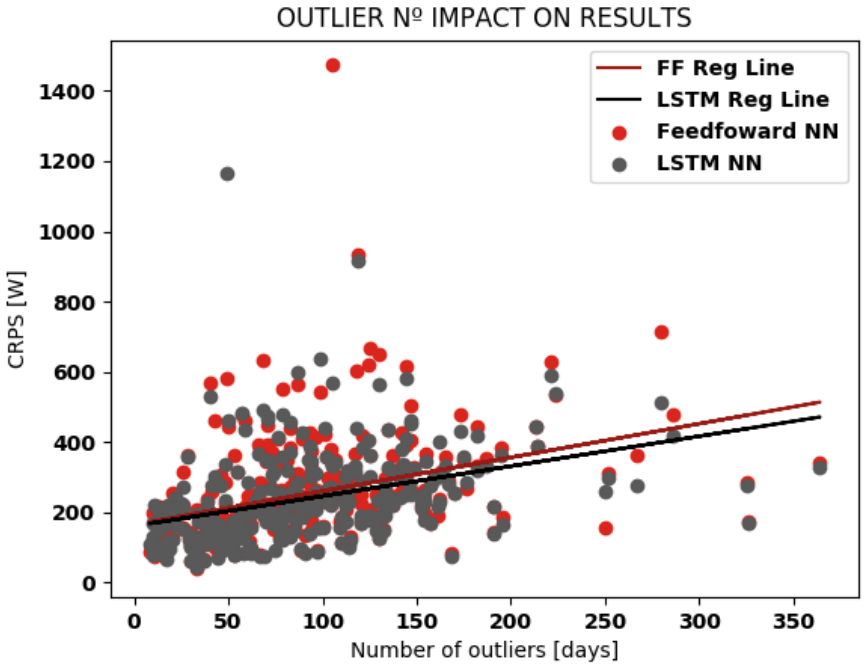


Figure 5.7: Behavioral pattern consistency impact results on the forecasting error of Ausgrid's household data

5.2.6 Impact study results review

Table 5.6 shows the results from the features' impact study. As explained in section 3.2.5, a significance level (α) of 5% is defined. That means that all the features with a p-value above 0.05 are considered non-significant. From the results it can be determined that the average power consumption, the presence of cycles, the

dispersion, and the pattern consistency significantly affects the forecasting accuracy. Furthermore, these features impact both feedforward and LSTM models.

Feature	**Feedforward NN**		**LSTM NN**	
	Pearson Coeff	p-value	Pearson Coeff	p-value
Average load	0.534	$1 \cdot 10^{-23}$	0.527	$1 \cdot 10^{-28}$
Cycles	0.572	$1 \cdot 10^{-27}$	0.554	$1 \cdot 10^{-25}$
Dispersion	0.201	$4 \cdot 10^{-4}$	0.223	$9 \cdot 10^{-5}$
Pattern complexity	0.044	0.45	0.054	0.35
Pattern consistency	0.371	$3 \cdot 10^{-11}$	0.369	$4 \cdot 10^{-11}$

Table 5.6: Overview over impact study results

6 Conclusion and Outlook

This thesis seeks to achieve two objectives, the first one it to develop a new forecasting model that improve short-term residential load forecasting. The second objective is to analyze which are the main smart meter data features that influence forecasting accuracy. Previous studies have shown that using machine learning algorithms can significantly improve the forecasting accuracy. Furthermore, human behavior seems to be the biggest factor of influence when forecasting household load data.

Long short-term memory (LSTM) recurrent neural network models have a memory cell capable of extracting long-term temporal relationships. This feature makes LSTM neural networks ideal for predicting smart meter loads as extracting complex trends could significantly improve the forecasting performance. Therefore, LSTM neural networks have been implemented in this thesis. The results from the LSTM model have been compared with the feedforward neural network model results obtained in [43].

Furthermore, the forecasting results largely depend on the selected data set. Choosing the optimal forecasting model is sometimes difficult due to the changes in the forecasting performance for different household data. In this work an impact study is conducted to analyze which features have an impact on forecasting accuracy.

There are three main conclusions extracted from the forecasting results of the feedforward and LSTM neural network models. The first one is that both models generate reliable load forecasts. The accuracy results obtained from the feedforward and the LSTM neural network models outperformed the results obtained from the persistence method, used as a benchmark method.

The second conclusion is that LSTM neural network models achieve generally better forecasting performance than feedforward models. The LSTM neural networks seem to better extract past trends than feedforward neural networks.

The third conclusion is that training LSTM neural networks requires more time than training feedforward neural networks. The computational requirements for training LSTM memory neural networks are higher than for training feedforward networks as a result, training a LSTM neural network model takes on average 20 times longer than training a feedforward model for the same data set.

Two conclusions are extracted from the smart meter features' impact study. First, average power consumption, presence of cycles, data dispersion and behavioral pattern consistency seem to have a significant impact on forecasting accuracy. Although these correlations are not evident when analyzing the point distribution, the correlation coefficients obtained show that there are linear correlations between these features and the forecasting accuracy. The average power consumption, presence

of cycles and data dispersion negatively affect forecasting accuracy. On the other hand, an increasing pattern consistency, reflected by a decreasing random behavior, tend to give better forecasting results.

The second main conclusion from the impact study is that the feedforward and the LSTM neural network models are affected by the same data set features.

This field of research is still relatively young and its changing at a very high pace. Therefore, there is space for research in many different areas, here three possible improvements to this work are proposed.

The first possible improvement consists on redefining the human behavior pattern extraction. The impact study presented have tried to extract two features from human behavioral patterns, the pattern complexity and the pattern consistency. The pattern complexity was defined by adding the daily profiles from the biggest k-mean cluster. However, extracting the number of clusters by using DBSCAN clustering could be a better way of defining the complexity of behavioral patterns. Furthermore, it is expected that the benefits of using LSTM neural networks are more important if the household's trends are complex.

The second improvement proposed is to individually tune each smart meter forecasting model before performing the data sets' feature impact study. In this work all forecasting model parameters were kept constant. Although the overall results were better than the benchmark methods, there were households with bad forecasting performance. Due to the households' different load behavior, forecasting model parameters should be individually tuned instead of kept constant.

The final improvement proposed consists on conducting an impact study on multiple data sets. The impact study was conducted on all the households from the Ausgrid data set. The analyzed households were all from the same geographical region and had rooftop panels installed. Therefore, an impact study on a more diverse set of household data set might give different results than the ones extracted in this work.

Appendix

A Appendix

A.1 Data set characterization example

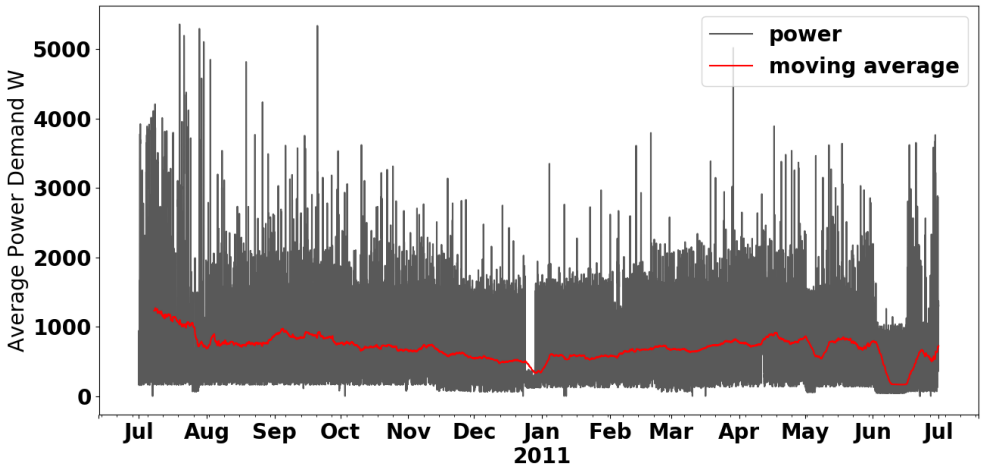


Figure A.1: Smart meter data from household 1

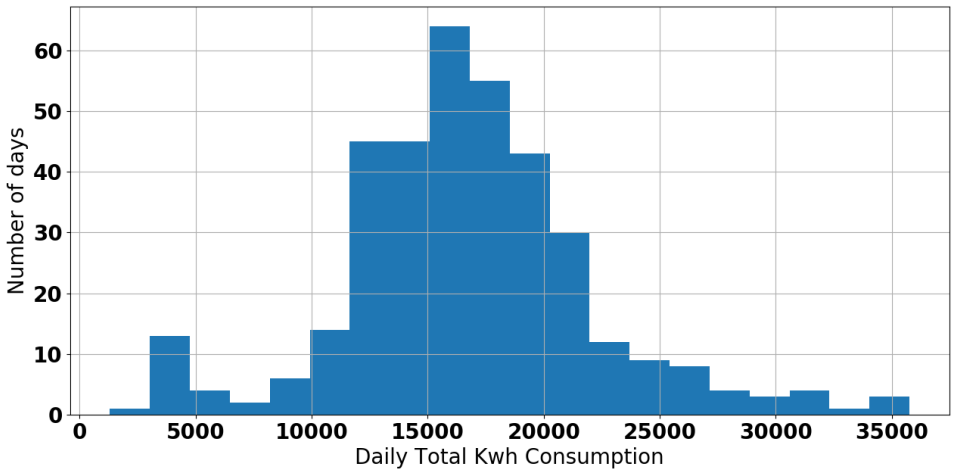


Figure A.2: Daily energy consumption dispersion in household 1

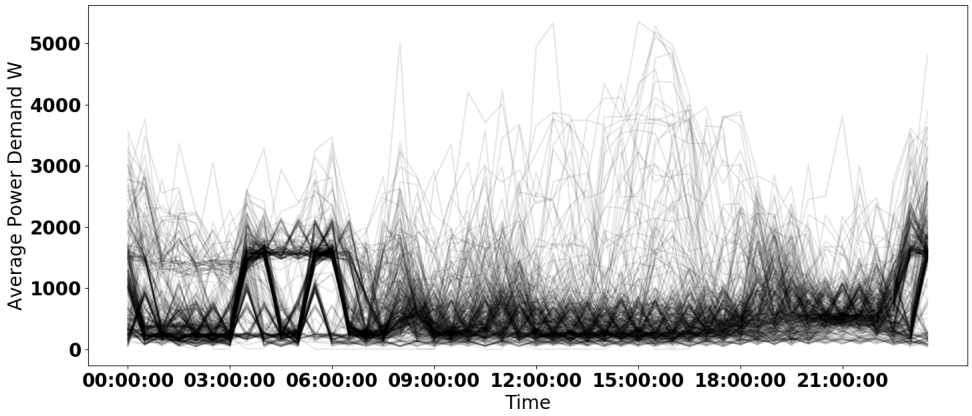


Figure A.3: Daily load profiles in household 1

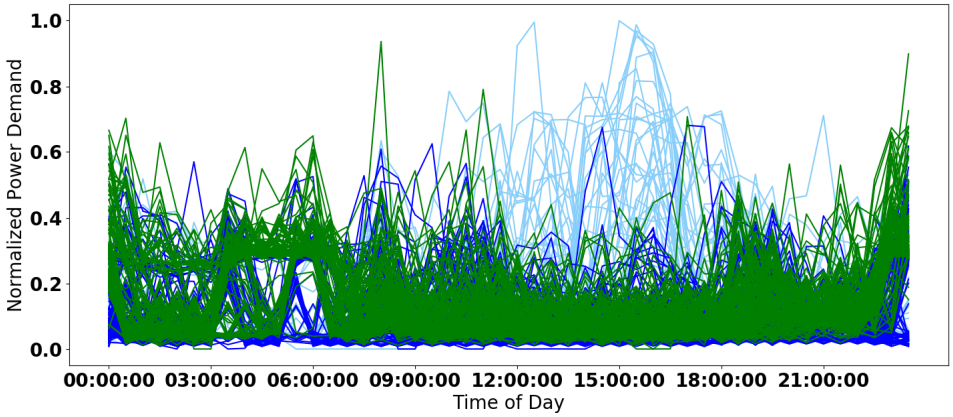


Figure A.4: Household 1 k-means clustering results

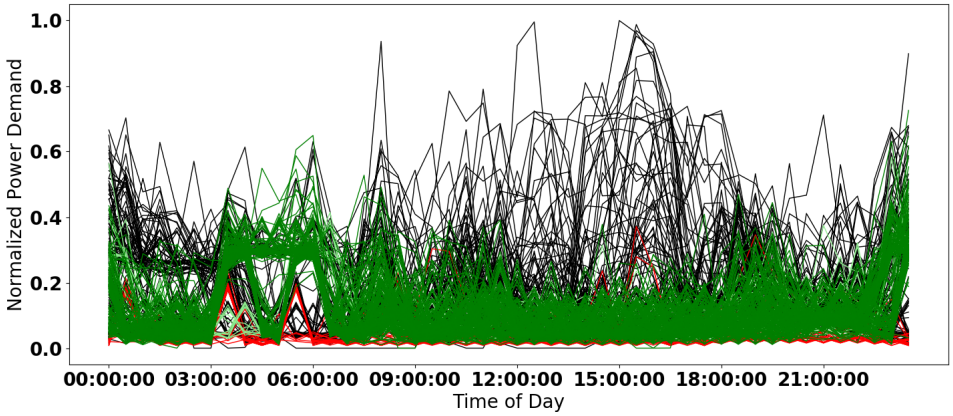


Figure A.5: Household 1 DBSCAN clustering results

A.2 Model comparison

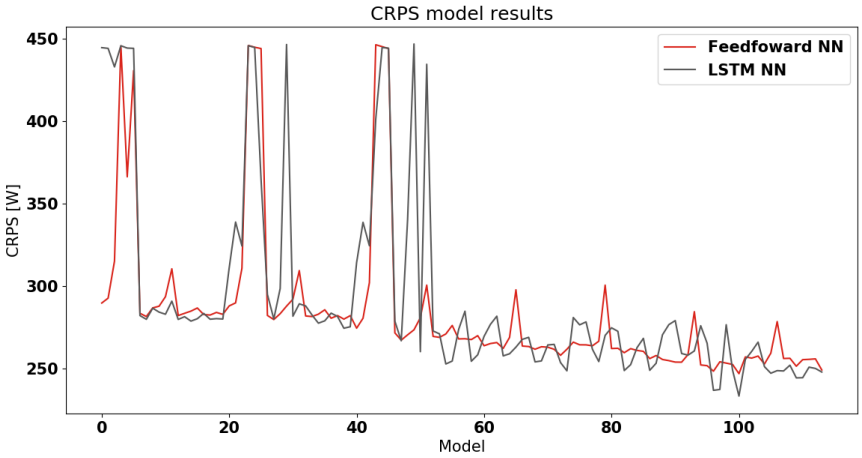


Figure A.6: CRPS results comparison from UCI data set grid search

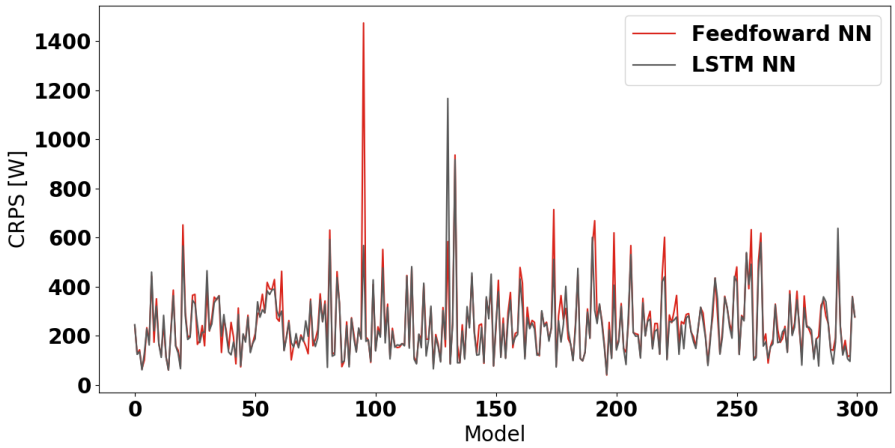


Figure A.7: Ausgrid CRPS results

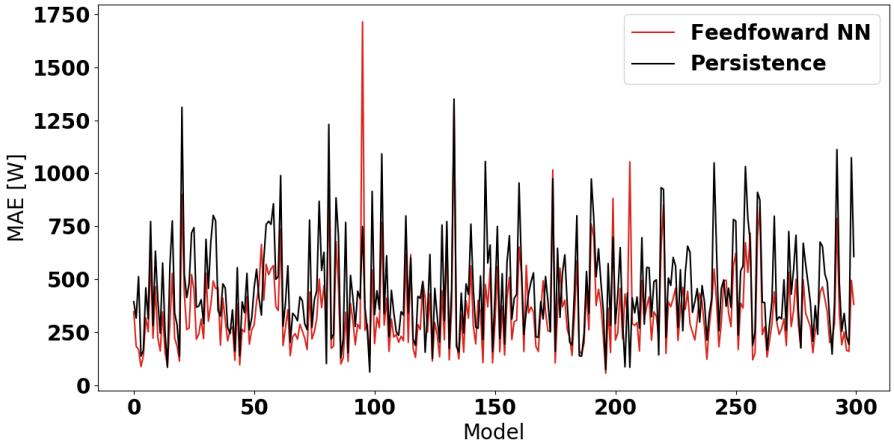


Figure A.8: Ausgrid feedforward MAE results

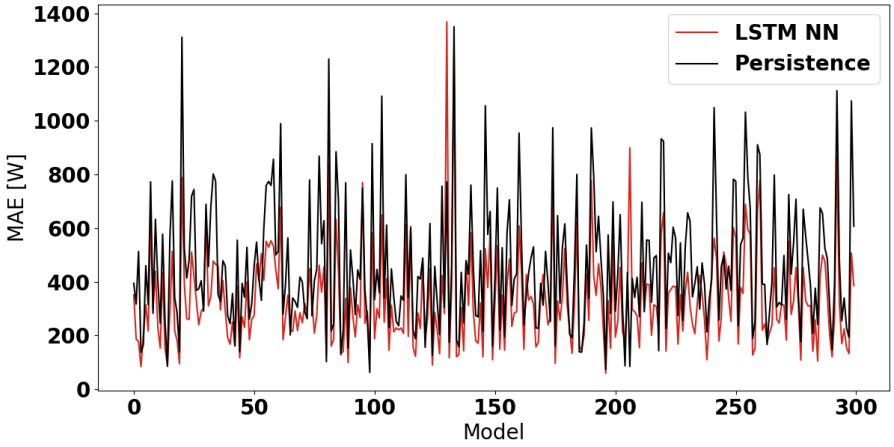


Figure A.9: Ausgrid LSTM MAE results

Acronyms

AI Artificial Intelligence

ANN Artificial Neural Network

CPU central processing unit

CRPS Continuous Ranked Probability Score

DBSCAN Density Based Spatial Clustering of Application with Noise

GPU graphics processing unit

HEMS home energy management system

LSTM Long Short Term Memory

MAE Mean Absolute Error

MDN Mixture Density Network

NLL negative log-likelihood

NN Neural Network

PDF Probability Density Function

PLF Probabilistic Load Forecasting

RNN Recurrent Neural Network

SDN Softmax Distribution Network

STLF Short Term Load Forecasting

List of Figures

List of Tables

Bibliography

[1] Siddharth Arora and James W. Taylor. "Forecasting electricity smart meter data using conditional kernel density estimation". In: *Omega*. Business Analytics 59 (Mar. 2016), pp. 47–59. ISSN: 0305-0483. DOI: 10.1016/j.omega.2014.08.008. URL: http://www.sciencedirect.com/science/article/pii/S0305048314001546 (visited on 02/27/2018).

[2] B. Asare-Bediako, W. L. Kling, and P. F. Ribeiro. "Day-ahead residential load forecasting with artificial neural networks using smart meter data". In: *2013 IEEE Grenoble Conference*. June 2013, pp. 1–6. DOI: 10.1109/PTC.2013.6652093.

[3] S. Balluff, J. Bendfeld, and S. Krauter. "Short term wind and energy prediction for offshore wind farms using neural networks". In: *2015 International Conference on Renewable Energy Research and Applications (ICRERA)*. Nov. 2015, pp. 379–382. DOI: 10.1109/ICRERA.2015.7418440.

[4] Sean Barker et al. *Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes*.

[5] Y. Bengio, P. Simard, and P. Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2 (Mar. 1994), pp. 157–166. ISSN: 1045-9227. DOI: 10.1109/72.279181.

[6] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, Inc., 1995. ISBN: 0198538642.

[7] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. en. 2nd ed. Springer Texts in Statistics. New York: Springer-Verlag, 2002. ISBN: 978-1-4757-7750-5. URL: //www.springer.com/us/book/9781475777505 (visited on 02/27/2018).

[8] K. Chen, Y. Zhou, and F. Dai. "A LSTM-based method for stock returns prediction: A case study of China stock market". In: *2015 IEEE International Conference on Big Data (Big Data)*. Oct. 2015, pp. 2823–2824. DOI: 10.1109/BigData.2015.7364089.

[9] Gianfranco Chicco. "Overview and performance assessment of the clustering methods for electrical load pattern grouping". In: *Energy*. 8th World Energy System Conference, WESC 2010 42.1 (June 2012), pp. 68–80. ISSN: 0360-5442. DOI: 10.1016/j.energy.2011.12.031. URL: http://www.sciencedirect.com/science/article/pii/S0360544211008565 (visited on 03/19/2018).

[10] *EISPC Update: Electric Load Forecasting Finds Its New Life in the Smart-Grid Era*. Tech. rep. 2015. URL: http://blog.drhongtao.com/2015/03/eispc-naruc-electric-load-forecasting.html (visited on 02/27/2018).

[11] G. Box, G. Jenkins, and G. Reinsel. *Time Series Analysis, Fourth Edition - Box - Wiley Online Library*. URL: http://onlinelibrary.wiley.com/book/10.1002/9781118619193 (visited on 02/27/2018).

[12] Krzysztof Gajowniczek and Tomasz Ząbkowski. "Short Term Electricity Forecasting Using Individual Smart Meter Data". In: *Procedia Computer Science*. Knowledge-Based and Intelligent Information & Engineering Systems 18th Annual Conference, KES-2014 Gdynia, Poland, September 2014 Proceedings 35 (Jan. 2014), pp. 589–597. ISSN: 1877-0509. DOI: 10.1016/j.procs.2014.08.140. URL: http://www.sciencedirect.com/science/article/pii/S1877050914011053 (visited on 02/27/2018).

[13] Alexis Gerossier et al. "PROBABILISTIC DAY-AHEAD FORECASTING OF HOUSEHOLD ELECTRICITY DEMAND". In: June 2017.

[14] M. Ghofrani et al. "Smart meter based short-term load forecasting for residential customers". In: *2011 North American Power Symposium*. Aug. 2011, pp. 1–5. DOI: 10.1109/NAPS.2011.6025124.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[16] Alex Graves and Navdeep Jaitly. "Towards End-To-End Speech Recognition with Recurrent Neural Networks". en. In: *International Conference on Machine Learning*. Jan. 2014, pp. 1764–1772. URL: http://proceedings.mlr.press/v32/graves14.html (visited on 02/27/2018).

[17] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[18] S. Hochreiter et al. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies". In: *A Field Guide to Dynamical Recurrent Neural Networks*. Ed. by S. C. Kremer and J. F. Kolen. IEEE Press, 2001.

[19] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[20] T. Hong. "Short Term Electric Load Forecasting". In: (Jan. 2010). ISSN: 978-1-124-47772-5.

[21] Eric Jones, Travis Oliphant, Pearu Peterson, et al. *SciPy: Open source scientific tools for Python*. [Online; accessed <today>]. 2001–. URL: http://www.scipy.org/.

[22] Jack Kelly and William Knottenbelt. "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes". en. In: *Scientific Data* 2 (Mar. 2015), p. 150007. ISSN: 2052-4463. DOI: 10.1038/sdata.2015.7. URL: https://www.nature.com/articles/sdata20157 (visited on 03/14/2018).

[23] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2014). arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980.

[24] W. Kong et al. "Short-Term Residential Load Forecasting based on LSTM Recurrent Neural Network". In: *IEEE Transactions on Smart Grid* PP.99 (2017), pp. 1–1. ISSN: 1949-3053. DOI: 10.1109/TSG.2017.2753802.

[25] W. Kong et al. "Short-Term Residential Load Forecasting Based on Resident Behaviour Learning". In: *IEEE Transactions on Power Systems* 33.1 (Jan. 2018), pp. 1087–1088. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2017.2688178.

[26] Roman Krzysztofowicz. "The case for probabilistic forecasting in hydrology". In: *Journal of Hydrology* 249.1 (Aug. 2001), pp. 2–9. ISSN: 0022-1694. DOI: 10.1016/S0022-1694(01)00420-6. URL: http://www.sciencedirect.com/science/article/pii/S0022169401004206 (visited on 02/27/2018).

[27] Peter Lusis et al. "Short-term residential load forecasting: Impact of calendar effects and forecast granularity". In: *Applied Energy* 205 (Nov. 2017), pp. 654–669. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2017.07.114. URL: http://www.sciencedirect.com/science/article/pii/S0306261917309881 (visited on 02/27/2018).

[28] Daniel Marino, Kasun Amarasinghe, and Milos Manic. "Building Energy Load Forecasting using Deep Neural Networks". In: Oct. 2016. DOI: 10.1109/IECON.2016.7793413.

[29] D. M. Minhas, R. R. Khalid, and G. Frey. "Short term load forecasting using hybrid adaptive fuzzy neural system: The performance evaluation". In: *2017 IEEE PES PowerAfrica*. June 2017, pp. 468–473. DOI: 10.1109/PowerAfrica.2017.7991270.

[30] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[31] Fotios Petropoulos et al. "'Horses for Courses' in demand forecasting". In: *European Journal of Operational Research* 237.1 (Aug. 2014), pp. 152–163. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2014.02.036. URL: http://www.sciencedirect.com/science/article/pii/S0377221714001714 (visited on 02/28/2018).

[32] Cesar Roda et al. "Home Energy Management System". In: (May 2014).

[33] J. Schlageter. "Electrical Load Forecast: Comparison and evaluation of different methods at a single-houshold level". Bachelor Thesis. 2015. URL: https://drive.google.com/file/u/0/d/0B6kVnEC2rBsTMXBWNnpQalFqRm8/view?usp=drive_web&usp=embed_facebook (visited on 02/27/2018).

[34] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: 15 (June 2014), pp. 1929–1958.

[35] Carl-Axel S. Staël Von Holstein. "Probabilistic forecasting: An experiment related to the stock market". In: *Organizational Behavior and Human Performance* 8.1 (Aug. 1972), pp. 139–158. ISSN: 0030-5073. DOI: 10.1016/0030-5073(72)90041-4. URL: http://www.sciencedirect.com/science/article/pii/0030507372900414 (visited on 02/27/2018).

[36] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". In: *arXiv:1409.3215 [cs]* (Sept. 2014). arXiv: 1409.3215. URL: http://arxiv.org/abs/1409.3215 (visited on 02/27/2018).

[37] S. Fan T. Hong. "Probabilistic electric load forecasting: A tutorial review". In: *International Journal of Forecasting* (2015). URL: https://drive.google.com/file/u/0/d/0B6kVnEC2rBsTaG5oUTNvdnM0Q0E/view?usp=drive_web&usp=embed_facebook (visited on 02/27/2018).

[38] S. Ben Taieb et al. "Forecasting Uncertainty in Electricity Smart Meter Data by Boosting Additive Quantile Regression". In: *IEEE Transactions on Smart Grid* 7.5 (Sept. 2016), pp. 2448–2455. ISSN: 1949-3053. DOI: 10.1109/TSG.2016.2527820.

[39] *UCI Machine Learning Repository: Individual household electric power consumption Data Set*. URL: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption (visited on 03/14/2018).

[40] R. E. Uhrig. "Introduction to artificial neural networks". In: , *Proceedings of the 1995 IEEE IECON 21st International Conference on Industrial Electronics, Control, and Instrumentation, 1995*. Vol. 1. Nov. 1995, 33–37 vol.1. DOI: 10.1109/IECON.1995.483329.

[41] O. Vinyals et al. "Show and tell: A neural image caption generator". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, pp. 3156–3164. DOI: 10.1109/CVPR.2015.7298935.

[42] Luciano Viola. "Segmentation of household load-profiles with K-means clustering algorithm". In: (July 2016).

[43] Julian Vossen. "Probabilistic forecasting of household electrical load using artificial neural networks". Bachelor Thesis. 2017. URL: https://drive.google.com/file/u/0/d/0B6kVnEC2rBsTU3hBSGllYnJVeG8/view?usp=drive_web&usp=embed_facebook (visited on 02/27/2018).

[44] C. Wan et al. “Probabilistic Forecasting of Wind Power Generation Using Extreme Learning Machine”. In: *IEEE Transactions on Power Systems* 29.3 (May 2014), pp. 1033–1044. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2013.2287871.

[45] Y. Yang et al. “A CFCC-LSTM Model for Sea Surface Temperature Prediction”. In: *IEEE Geoscience and Remote Sensing Letters* 15.2 (Feb. 2018), pp. 207–211. ISSN: 1545-598X. DOI: 10.1109/LGRS.2017.2780843.