Universidad de Alcalá
Escuela Politécnica Superior

Máster Universitario en Ingeniería Industrial

Trabajo Fin de Máster

Designing Artificial Neural Networks (ANNs) for Electrical Appliance Classification in Smart Energy Distribution Systems

Author: Laura de Diego Otón

Advisor: Álvaro Hernández Alonso

Co-advisor: Rubén Nieto Capuchino

2022

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Máster Universitario en Ingeniería Industrial

Trabajo Fin de Máster

Designing Artificial Neural Networks (ANNs) for Electrical Appliance Classification in Smart Energy Distribution Systems

Author: Laura de Diego Otón

Advisor: Álvaro Hernández Alonso

Co-advisor: Rubén Nieto Capuchino

Tribunal:

President: Ignacio Bravo Muñoz

1st Vocal: Carlota Salinas Maldonado

2nd Vocal: Álvaro Hernández Alonso

Deposit date: 22/July/2022

"Satisfaction lies in the effort, not in the attainment, full effort is full victory"

Mahatma Gandhi

Acknowledgements

First and foremost, I would like to recognize and give my warmest thanks to my supervisors Álvaro Hernández and Rubén Nieto who made this work possible. Their guidance, support and advice carried me through the process of researching and writing this thesis. Moreover, this effort would not have been possible without the support of the PoM-UAH and MICROCEBUS projects, which funded this research.

Words cannot express my gratitude to my family for their inspiration, love, and support. Without their tremendous understanding and encouragement over the past few years, it would have been impossible for me to complete my studies and neither this project.

Last but not least, I would like to express my deepest gratitude to my friends, my master's classmates, and my research team, especially to my lab mates, for all the entertainment and emotional support, but also for making me see that I must be more aware of my skills and achievements.

This accomplishment would not have been possible without all of them. Thank you.

Resumen

En este proyecto se abordará el problema de la desagregación del consumo eléctrico a través del diseño de sistemas inteligentes, basados en redes neuronales profundas, que puedan formar parte de sistemas más amplios de gestión y distribución de energía. Durante la definición estará presente la búsqueda de una complejidad computacional adecuada que permita una implementación posterior de bajo costo. En concreto, estos sistemas realizarán el proceso de clasificación a partir de los cambios en la corriente eléctrica provocados por los distintos electrodomésticos. Para la evaluación y comparación de las diferentes propuestas se hará uso de la base de datos BLUED.

Palabras clave: Monitorización de Cargas No Intrusiva, Desagregación de Energía, Red Neuronal Profunda, Eficiencia Energética, Contadores inteligente, Identificación de Electrodomésticos.

Abstract

This project will address the energy consumption disaggregation problem through the design of intelligent systems, based on deep artificial neural networks, which would be part of broader energy management and distribution systems. The search for adequate computational complexity that will allow a subsequent implementation of low cost will be present during algorithm definition. Specifically, these systems will carry out the classification process based on the changes caused by the different appliances in the electric current. For the evaluation and comparison of the different proposals, the BLUED database will be used.

Keywords: Non-Intrusive Load Monitoring, Energy Disaggregation, Artificial Neural Network, Energy Efficiency, Smart Meter, Appliance Identification.

Extended summary

Reduction of energy consumption is one of the main focuses of interest of society in recent years. Regarding this problem, intelligent electricity distribution systems seek to manage consumption more efficiently. Likewise, the incorporation of technology in homes allows consumers to be informed about the specific consumption of each of the devices they have connected to the electricity grid. With this information, applications can be defined for the creation of consumption information reports, control of electrical parameters, management of consumption, and generation of alarms.

In practice, there are two main ways of extracting this information: through the individual monitoring of each of the loads, which is more intrusive and expensive, and due to this, it is not usually the most accepted alternative; or by using Non-Intrusive Load Monitoring techniques (NILM), which obtains this information from the user's aggregate energy signal captured by a Smart Meter (SM). In the field of these non-intrusive techniques, the extraction of the individual profiles of the main household appliances from the consumption profile is carried out by intelligence systems, such as Artificial Neural Networks (ANNs), among others.

This is the place where this Master's Thesis takes part. This works presents three different solutions which are designed with the aim of classifying the different household appliances based on the on/off events which are extracted from the electrical current signal acquired at high frequency.

The proposed architectures have a first stage, where the input samples are adapted to meet some requirements; and a second and final stage that particularly deals with the classification of the load by two different topologies of ANNs: a recurrent network for the first, and a convolutional network for the second approach. Different types of features from the samples are considered either from time or frequency domain.

Experimental results show that the defined algorithms achieve a suitable classification performance for sixteen or seventeen appliances (or groups of appliances). This validation was carried out by using real data obtained from the public Building-Level fUlly labelled Electricity Disaggregation database (BLUED). However, the values obtained are also compared bearing in mind the computational complexity presented by the solutions. Finally, taking into account both criteria, the second architecture combines an adequate classification performance, of around 90 % for the F1 score, with the less complex network.

Additionally, an analysis of future actions required to carry out the optimization of the proposal and some possible future works related to the continuation of the project will be included.

Contents

List of Figures

List of Tables

List of Acronyms

Adam	Adaptative Moment Estimation
ADL	Activities of Daily Living
AFE	Analog Front-Ends
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
BLUED	Building-Level fUlly labelled Electricity Disaggregation
BRNN	Bidirectional Recurrent Neural Network
CNN	Convolutional Neural Network
dB	Decibels
dBW	Decibels watt
DNN	Deep Neural Network
FFT	Fast Fourier Transform
FP	False Positive
FPGA	Field-Programmable Gate Array
FN	False Negative
FSM	Finite-State-Machine
GEINTRA	Group of Electronic Engineering Applied to intelligent Spaces and Transport
GRU	Gated Recurrent Unit
HEMS	Home Energy Management Systems
HES	Household Electricity Survey
HMM	Hidden Markov Models
kNN	k-Nearest Neighbours
LR	Logistic Regression
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NILM	Non-intrusive load monitoring
PCA	Principal component analysis
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
SNR	Signal-to-Noise Ratio
SM	Smart Meter
SoC	System-on-Chip
Sps	Samples per second
STFT	Short Time Fourier Transform
SVM	Support Vector Machine
TP	True Positive
TN	True Negative

Chapter 1 Introduction

At present, energy systems are evolving into what is known as smart grids, an electrical network that employs innovative intelligent monitoring, control, communication, and self-healing technologies. One of the crucial components of the smart grid (O. Majeed Butt et al., 2021), in many developing countries, is the smart home and its control systems, the so-called Home Energy Management Systems (HEMS) (B. Mahapatra & A. Nayyar, 2019). Their contribution is the optimization of energy consumption, allowing substantial savings by users, and boosting the consistency of the electricity grid by permitting the consumption of household appliances to be adjusted according to the demand for electrical energy, which is being produced in the electricity system.

Information on energy usage is obtained thanks to the Advanced Metering Infrastructure (AMI) which interacts with the Smart Meters (SM) of homes (J. Zheng et al., 2013). Within an electrical context, SM are devices that record electrical measurements, such as voltage and current levels, and communicate this data to electricity suppliers and clients. Regarding the improvement of the management of consumer consumption behaviour, it may be interesting to know the individual consumption of each appliance. For that purpose, Non-Intrusive Load Monitoring (NILM) systems are defined to identify each electrical device through the processing of the load signals acquired at a single point at the entrance of the house; in other words, without the need to connect individual monitors on each home device (G. Bucci et al., 2021; G. F. Angelis et al., 2022; Y. Himeur et al., 2022).

Several NILM algorithms include a classification stage which analyses the characteristics of interest of the input samples to determine which device is working at each time instant. In this scope, there are some approaches based on advanced machine learning techniques of great interest at this moment, such as Artificial Neural Networks (ANNs). As will be explained later, ANNs are data processing models, based on biological neural networks, capable of "learning" through a feedback process that compares the obtained result with the desired one.

Specifically, as illustrated in Figure 1, this project seeks to design and evaluate intelligent systems based on different ANN topologies for the classification of the main domestic loads from the aggregate signals provided by the SM with the purpose of using them in intelligent energy management and distribution systems.

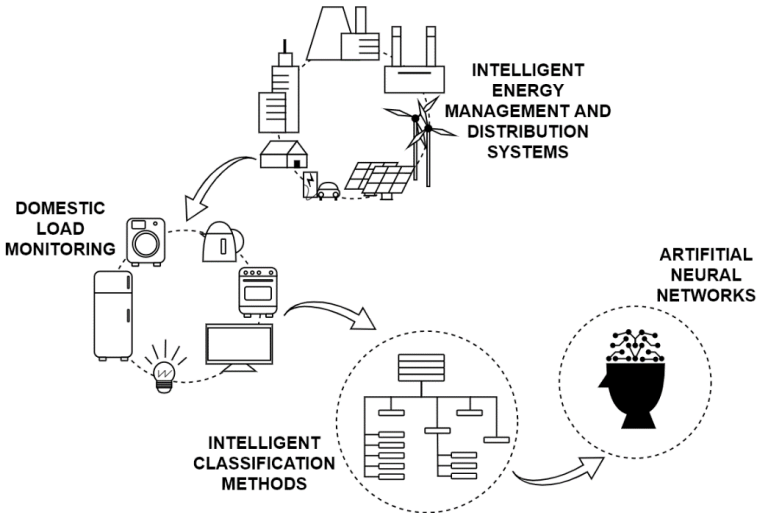


Figure 1. Scope of the Master's Thesis.

1.1 Context

Essential knowledges included in the University Master's Degree in Industrial Engineering have been required for the realization of the project. These skills are related to subjects of the specialty of Smart Energy Generation and Distribution, such as Introduction to Smart Energy grids or Distributed Generation and Network Quality. Additionally, this work was carried out in the GEINTRA research group, belonging to the Department of Electronics, within projects granted through the Ministry of Science, Innovation and Universities. Namely, the PoM-UAH project (PID2019-105470RA-C33), where the ultimate goal is to improve and promote the independent life of people with dementia, mild cognitive impairment, or people with psychological disorders through the use of localization techniques and minimally invasive systems; and the MICROCEBUS project (RTI2018-095168-B-C51), focused on indoor positioning systems.

1.2 Objectives

The proposal is focused on the evaluation of different ANN topologies for the electrical appliances classification stage in order to be part of a smart energy management and distribution system within the household. Nevertheless, several intermediate steps will be necessary until this final objective has been attained. Their individual aims are the following:

- Analyse previous works and backgrounds associated with the main topic, which will be used as the basis for the proposal's development.
- Configure the appropriate experimental framework, from the selection of the database with the required features to carry out the validation of the algorithms.
- Definition of intelligent systems made up of certain neural network topologies for the recognition of electrical household appliances within the energy disaggregation.
- Compare the classification results of the different alternatives to determine which one offers the highest performance and evaluate the feasibility of implementation on platforms in real time.

1.3 Structure of the document

Down below, the structure of the remaining chapters is described to familiarize readers with the content of the document and help them have a better understanding of the project.

CHAPTER 2. BACKGROUND

This first part of the document will analyse the main concepts on which the project is based. Specifically, it will show not only what is meant by NILM techniques, but also their main fields of application and the impact of the issue. Additionally, a brief explanation of the different methods that exist these days for energy disaggregation will be provided. Finally, the data processing of algorithms based on neural networks will be studied.

CHAPTER 3. PROPOSED ARCHITECTURES

The alternatives developed for this project will be described in depth distinguishing between the neural network topology used for the classification architecture and the features of interest from the input samples.

CHAPTER 4. EXPERIMENTAL RESULTS

In this chapter, the specific characteristics of the database used for the validation of the proposed intelligent systems will be described. In addition, owing to the fact that several types of metrics should be used for the performance evaluation of the different models, their definition will be covered. Furthermore, the setup for the appraisal will be specified. After all, the experimental results obtained through the different alternatives will be compared.

CHAPTER 5. CONCLUSION AND FUTURE WORKS

With reference to the conclusions, the value of the study and its contribution to the current state-of-the-art will be discussed. Secondly, not only certain improvements for the proposal presented in this thesis, but also several extensions in relation to the continuation of the research will be listed. Finally, the publications derived from this thesis will be commented.

1.4 Publications derived from this thesis

Thereupon, the publications derived from the research carried out during the period of development of this Master's Thesis are listed.

CONFERENCE PAPERS

- R. Nieto Capuchino, L. de Diego-Otón, Á. Hernández, J. Ureña. (2020). Finite Precision Analysis for an FPGA-based NILM Event-Detector. In the *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring (NILM'20)*, pp. 30–33. 10.1145/3427771.3427849.
- L. de Diego-Otón, Á. Hernández Alonso, R. Nieto Capuchino, & M. C. Pérez Rubio. (2021). Evaluación de una Arquitectura CNN para la Identificación de Cargas en NILM. In the *XXVIII Seminario Anual De Automática, Electrónica Industrial E Instrumentación (SAAEI'21)*, pp. 116-121.
- R. Nieto, L. de Diego-Otón, Á. Hernández, & J. Ureña. (2021). Data Collection and Cloud Processing Architecture Applied to NILM Techniques for Independent Living. In the *IEEE International Instrumentation and Measurement Technology Conference (I2MTC'21)*, pp. 1-6. 10.1109/I2MTC50364.2021.9460010.

- L. de Diego-Otón, Á. Hernández Alonso, R. Nieto Capuchino, & M. C. Pérez Rubio. (2022). Comparison of Neural Networks for High-Sampling Rate NILM Scenario. In the *IEEE International Symposium on Medical Measurements and Applications Proceedings (MeMeA'22)*.

- R. Nieto Capuchino, L. de Diego-Otón, Á. Hernández, J. Ureña. Design of a SoC Architecture for High-Sampling Non-Intrusive Load Monitoring Systems. In the *IEEE Transactions on Instrumentation and Measurement*. Submission expected in July 2022.

Chapter 2 Background

This chapter serves as a general presentation of the baseline scenario from which this project is established. Thus, the object of study is situated within a broader theoretical approach that includes the concept and applications of the NILM techniques and the operation of the algorithms used for the distinction of domestic loads, analysing more in depth the ANNs.

2.1 NILM techniques

As mentioned above, NILM systems are capable of identifying the specific device that is being used by processing the main consumption signals recorded by the SM in the house. In addition, because there is no need to connect individual monitors on each device, it can be said that NILM represents a low-cost and non-intrusive alternative. The ability to distinguish is essentially since, although two devices consume the same total power, they may have different impedances and the effects on voltage and current signals are also different. As can be observed in Figure 2, the procedure required to distinguish these events consists of several parts involving data acquisition, feature extraction and load disaggregation.

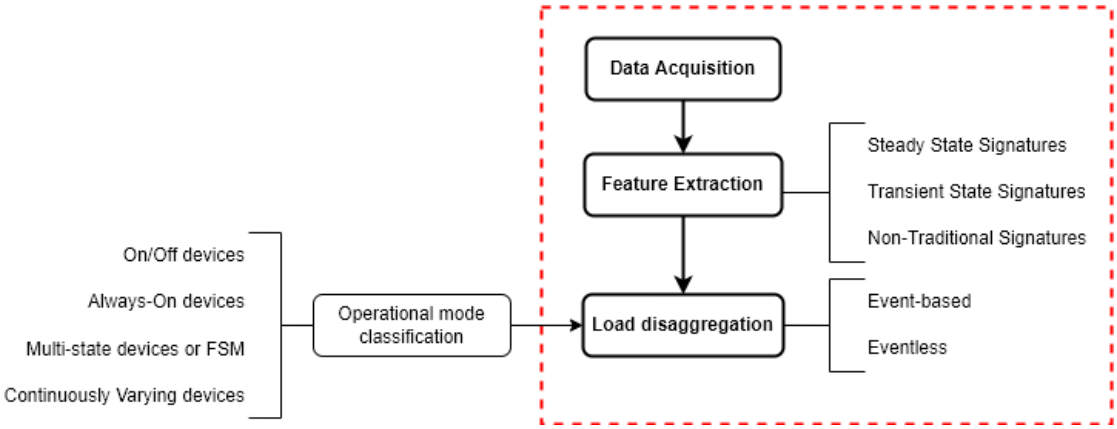


Figure 2. Relevant aspects of the parts of the NILM procedure.

Data collection is a key aspect that influences system-wide performance and defines which applications or challenges can be addressed. As discussed above, the SM performs this task by measuring the voltage and current consumed in homes or buildings. In this stage, determining the sampling frequency is crucial. Previous works (A. S. Siddiqui & A. M. Sibal, 2020) have already shown that sampling frequencies in the kHz range (high frequency) are more suitable for these techniques than rates around 1Hz (low frequency). The main reason is that high frequencies allow deeper analysis of measurements and therefore the extraction of more significant features, which can help achieve a better result since they make possible to distinguish loads with lower energy consumption. The features of interest can be classified depending on where they are extracted (J. Revuelta Herrero et al., 2018).

- **Steady State signatures**

As its name reveals, they are extracted during steady state. However, whether the sampling process is at high or low frequency, they can be information of waveform distortion or they can be power-related attributes, respectively.

- **Transient State signatures**

In this case, they are obtained from the transient state, and they have less similarity between appliances compared to the previous group. Nevertheless, to obtain them it is required high sampling rate.

- **Non-Traditional signatures**

These characteristics do not require large sample rates because they involve aspects such as the frequency of daily use or the time interval that the device can be statistically turned on.

The selection of one or another feature also depends on the approach of the disaggregation stage: event-based or eventless. First of all, in NILM an event is defined as the state transition of the aggregate electrical measurements. What differentiates these approaches is that:

- **Event-Based**

The method is focused on detecting events and classifying the appliances associated with the features of these events.

- **Eventless**

By contrast, these algorithms use the measured features in an attempt to infer the state of the different appliances that are operating at a given instant.

However, the problem can become more complicated since the number of operating states differs between the electronic devices. Commonly, four types can be distinguishing, whose functionality is represented in Figure 3:

- **On/Off devices**

Appliances characterized by consuming a constant level during their stable state of operation. Most household electrical devices are included in this category.

- **Always-On devices**

They also have an almost constant consumption but operate 24 hours.

- **Multi-state devices or Finite State Machines (FSM)**

In this group the devices have more than one active state and with different levels of consumption, typically on a rolling basis.

- **Continuously Varying devices**

The consumption of these devices varies over time but not on a rolling basis, consequently, the set of states is not possible to be disaggregated.

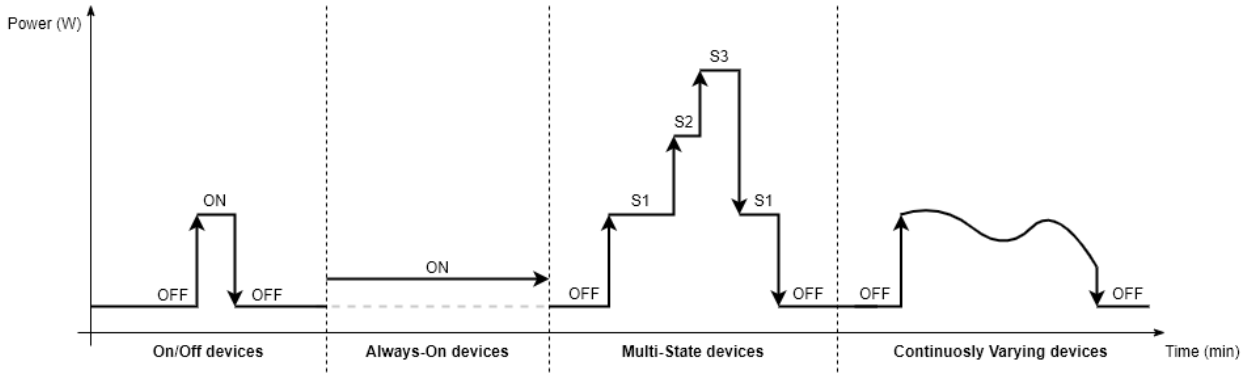


Figure 3. Classification of electronic devices depending on the operating states.

NILM techniques have some predominant applications directly related with the consumption (D. Christensen et al., 2012). For example, the most widespread application is to provide detailed bill information to users, which allows them to have some of the necessary tools to reduce their expenses. This information can also contribute to energy efficiency by allowing energy consumption to be optimized with simple actions such as avoiding unnecessary switching that produce unwanted consumption or selecting the appropriate time to use some specific devices whose consumption is determinist. In these terms, NILM is expected to have a significant impact in the coming years as the use of smart grids and demand response programs spreads. On the other hand, the monitoring of household appliances can be derived in a remote indirect health monitoring (A. Ruano et al., 2019; S. Dai et al., 2021; Y. H. Lin, 2022) since daily activities within the home are strongly connected to the use of some of these devices. From the measurements obtained by the SM can be inferred the ability of the person to perform what is called Activities of Daily Living (ADL), in other words, routine activities people do every day without assistance (S. Katz, 1983). This use case is and will be an intense topic of study due to the fact that the increase in life expectancy leads to a growth of the elderly population, and this part of the population requires special care to guarantee the best possible quality of life, especially in their own homes. Finally, other applications of these techniques linked to the previous points (B. Najafi et al., 2018) are occupancy detection which may be used by companies to offer an additional service without deploying any extra sensor; and illegal load detection that allows to identify possible energy thefts in both, public and private, buildings.

2.1 Appliances Identification methods

Once the type of disaggregation method has been selected and the features of interest have been extracted, it is the turn to define the algorithms used to determine the devices to which these operating characteristics belong. The process these intelligent systems use to distinguish objects is similar to that of humans. For example, the most common learning task is when a human or machine is trained to classify and, to do that, a set of samples along with their class labels are showed to them. However, the learning process can follow two principal approaches: supervised and unsupervised.

- **Supervised learning**

The algorithms make predictions about the training data by comparing it with the result and adjust their parameters until the correct answer is obtained.

- **Unsupervised learning**

These models work on their own to extract the unlabelled data structure.

In addition to this main different, and as a consequence, each approach has different principal tasks. For example, supervised learning is used for classification and regression problems, whereas unsupervised learning is applied to clustering or dimensionality reduction applications. Furthermore, unsupervised learning approach is greatly studied due to the fact that it requires minimal or no prior information; however, in general, compared to supervised disaggregation methods, they obtain lower accuracy. On the other hand, there is another approach that can be considered as an intermediate state between the previous ones, known as "semi-supervised", and that needs to train a little amount of data at the beginning of the process to perform the task for which it has been defined.

Hereinafter, the most widely applied techniques based on both supervised and unsupervised learning approaches, whose classification is represented in Figure 4, will be listed. In the case of the ANN, its description will be made in its own section due to its relevance for this Master's Thesis.

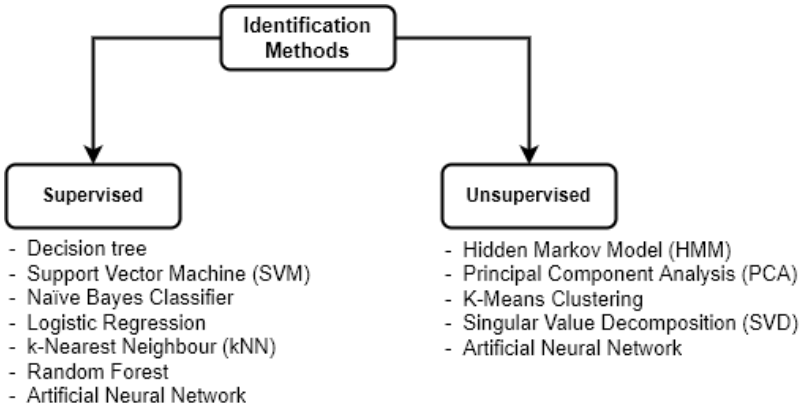


Figure 4. Classification of identification methods.

Firstly, the most relevant supervised learning methods will be described below:

- **Decision tree**

Decision tree learning belongs to the supervised learning methods (M. Somvanshi et al., 2016). It is a predictive model, used for regression and classification, that is among the most popular machine learning algorithms given its intelligibility. This model allows to predict the value of the target variable by learning decision rules inferred from the training data. Decision rules are usually based on simple comparisons between the values of the new attribute with the attribute of the record. Then, the continuous division of the data through the different decision nodes occurs to finally reach the decision sheets, which are the final outcomes. Within the framework of NILM, (T. -T. -H. Le et al., 2020) proposed a method called FFT-BDT. This structure is based on the steady-state characteristics extracted through the FFT process (*"Fast Fourier Transform"*), from the magnitude and phase of the electric current, which are used as the input of a bagging decision tree. Additionally, in (M. Nguyen et al., 2015) a decision tree is developed using the change in the measured power components calculated from the voltage and current signals.

- **Support Vector Machine**

Support Vector Machines (SVM) are a set of supervised learning methods used for classification, regression, and outliers' detection (M. Somvanshi et al., 2016). Being based on static learning frameworks of Vapnik-Chervonenkis theory (V. Cherkassky & F. Mulier, 1999), SVMs are known as the most robust prediction algorithms. In this context, data points are seen as p-dimensional vectors and the goal of these algorithms is to construct the hyperplane, in a high-dimensional space, that can separate the classes to

which these data points belong. The mappings used by SVM schemes are designed to ensure that dot products of input data vectors pairs may be computed easily by defining them in terms of a kernel function (A. Patle & D. S. Chouhan, 2013). Following this methodology, hyperplanes are defined as the set of points whose dot product with a vector in that space is constant. Generally, there are many hyperplanes that could achieve the final goal, however, the best of them is the one that implies the greatest margin or separation between the nearest data point of any class. Moreover, although they are mostly used for non-probabilistic linear binary tasks, SVMs are able to perform non-linear classification by using different kernel functions. In (A. S. Hernandez et al., 2021) a small-scale NILM system is defined where a SVM is used to extract and classify loads. Moreover, the proposed system in (Y. Lin & M. Tsai, 2011) uses a hierarchical SVM as a load identifier, which is able to identify the load operation state of single or multiple load operation scenarios.

- **Naïve Bayes Classifier**

Naïve Bayes classifiers are among the simplest and most effective Bayesian network models used for classification. The seed of these classifiers is Bayes' Theorem (P. Schulman, 1984), however, they assume that the value of each of the characteristics is independent of the value of any other, given the class variable. The procedure consists of using Bayes' Theorem to obtain the subsequent probability from the frequency data probabilities of given features. Furthermore, this type of classifiers requires a supervised learning training process and, although it could be useful for large datasets, it requires only a small number of samples to estimate the parameters necessary for classification. Regarding the NILM applications, (C. C. Yang et al., 2017) shows how the use of a Naïve Bayes classifier for the classification of disaggregated devices offers satisfactory results only for a single feature. In a more recent work, (P. Adjei et al., 2020), the disaggregation is obtained through the Gaussian Naïve Bayes classifier models based on the combination of appliances.

- **Logistic Regression**

Logistic regression (LR) (D. G. Kleinbaum & M. Klein, 2002), rather than being a regression model, is a statistical model used, above all, for binary classification problems. It is an effective supervised algorithm based on a logistic function, whose range is restricted between 0 and 1, and, compared to the linear algorithm, it does not require a linear relationship between the input variables and their predictions. The performing of this type of models is compared in (B. Bertalanic et al., 2021) with respect to other methods, such as the SVM, kNN, MLP and RF. However, in the context defined by the authors, the best results have been obtained with the SVM.

- **k-Nearest Neighbour**

The k-Nearest Neighbours (kNN) algorithm belongs to the supervised learning group, but it is non-parametric (K. Taunk et al., 2019). It is used for both regression and classification tasks. Classification problems are solved by calculating the distances between a query and the rest of the data points and obtaining the most frequent label for a specific number k of the nearest data points, called as neighbours. In the case of regression problems, the average of the labels is calculated. In both cases, choosing the right k value is a key factor and normally this process is done by trying several values and choosing the one with the best results. An application example of these algorithms for energy efficiency is presented in (C. C. Yang et al., 2018). In this work, satisfactory results are obtained for both classifiers, kNN and Naïve Bayes, using a single feature and a combination of an ON-OFF based approach and the goodness-of-fit technique for event detection.

- **Random Forest**

Random forest (RF), also called as random decision forest, is an ensemble supervised learning method for applications such as classification and regression (L. Breiman, 2001). It is built by multitude of decision trees combined by using the bagging or bootstrap aggregation ensemble technique. This method creates different subsets from the training data samples and the final result is based on the majority vote for classification or averaging for regression, as in the previous algorithm. In (Z. Xiao et al., 2021) a specific NILM application for disaggregating the cooling loads is presented. The method used is based on RF, although two approaches are defined depending on the availability of weather data. In a general way, (Wu et al., 2019) proposes a classification method which also uses RF as a learning algorithm. In this case, the algorithm outperforms other classification results obtained in previous works.

Second, the best-known unsupervised learning methods will be presented:

- **Hidden Markov Models**

Hidden Markov Models (HMM) are statistical algorithms based on the Markov chain, which is a model capable of offering information about the probabilities of sequences of random variables, also called states or events. Furthermore, the Markov chain assumes that all that matters is the current state if the goal is to predict the future in the sequence, in other words, states before the current state has no impact on the future. However, the functionality of Markov chain is limited due to the fact that, in many cases, the states of interest are hidden. For instance, in NILM the visible sequence would be the aggregate consumption and the hidden events, the states of each device. HMMs allow working with observed events and hidden events, which are considered causal factors in the probabilistic model. In addition, for this type of models that contain hidden variables, it is critical to determine which sequence of variables is the underlying source of some sequence of observations. This task is called decoding task and algorithms such as Viterbi's algorithm (G. D. Forney, 1973) are commonly used to perform it. In (H. Kim, 2011) an existing factorial HMM model and other three new models are used, with low-frequency measurements, with the aim of defining an effective method that would facilitate the conservation of electricity in residential environments.

- **Principal Component Analysis**

Principal component analysis (PCA) is an unsupervised non-parametric statistical algorithm commonly used in a wide variety of applications. It is based on the process of calculating the Principal Components (M. Bilodeau & D. Brenner, 1999), which can describe the data points using fewer dimensions. The main components of a set of data points are calculated vectors that, for example, explain the greatest amount of variance in the original data. This original data can be represented as features vectors and the PCA allows to represent this data as linear combinations of principal components. In the context of NILM, these types of algorithms are generally used to display high-dimensional data in a low-dimensional space or feature subset by preserving maximum information of the initial data, such as in (A. Moradzadeh et al., 2020) or (S. Qi et al., 2021).

- **K-Means clustering**

K-means clustering is one of the simplest unsupervised algorithms (S. Na et al., 2010). The method works as follows: the first step is to randomly select some data points, which will be called centroids; with these initial points for each cluster, an iterative process of calculating the optimized positions of the centroids will be carried out; the process stops when the centroids have stabilized, or the defined number of

iterations has been reached. By using this method, in (P. G. Papageorgiou et al., 2021), it was possible to form unlabelled datasets from information on current amplitudes for the first three odd harmonic orders produced by different combinations of devices operating simultaneously. Furthermore, in (A. Yasin & S. A. Khan, 2018), by combining this methodology with and event detector and an on-off paring stage, a complete NILM system is proposed.

- **Singular Value Decomposition**

Singular Value Decomposition (SVD) (X. Li et al., 2019) is the last of the unsupervised learning algorithms detailed in this section. Essentially, SVD is the factorization of a matrix into three matrices and is used to extract the most relevant features from the original matrix containing the vectorized data. In order to use this method to solve classification or regression problems, some other algorithms are used to complete the process. This may be why this type of learning method has not been widely used for energy disaggregation in the application of NILM.

2.2 Artificial Neural Networks

Artificial Neural Networks are a subset of machine learning which is at the heart of deep learning. As could be observed in Figure 4, ANNs could be trained with a supervised and unsupervised learning process. However, before talking about the learning process, it should be clear how they work and what their structure is. As their name suggests, its structure is inspired by the human brain and mimics the way biological neurons connect to each other. These neurons will be the fundamental units of the structure. In this context, each of these units processes a specific input, by using a particular function, and the extracted information is sent to the next neuron. These neurons are arranged in layers where they work together and, consequently, the typical architecture of RNAs contains an input layer, one or more hidden layers, and an output layer. The learning process is carried out by adjusting the weights and bias that are associated with the connections between neurons, which makes it possible for it to adapt to the particular problem at hand. The training process relies on improving the accuracy over time. The evaluation of this accuracy is done by using a cost-loss function. The ultimate goal is to minimize this cost function hence the model makes adjustments through gradient descent to reach the point of convergence or local minimum.

Neural networks can be classified into different types depending on the purpose. However, the following types are the most common architectures:

- **Perceptron**

The Perceptron is the oldest and simplest one, created by Frank Rosenblatt in 1958 (F. Rosenblatt, 1958), and is has a single neuron. It is an algorithm used for binary classifications trained by supervised learning. In practice, these types of neural networks are replaced by the structures detailed below.

- **Multi-Layer Perceptron**

The Multi-Layer Perceptron (MLP), also called feedforward neural network, consists of multiple layers, where they are interconnected in such a way that each neuron in a layer has direct connections to the neurons of the following layer and there are no feedback connections. MLP uses a supervised learning technique for training called backpropagation, which, once the results' error has been obtained, travels back from the output layer to the hidden layers to adjust the weights in such a way that the error decreases. The process keeps repeating until the desired result is achieved. In the field of NILM, one of the most recent works (S. Yaemprayoon & J. Srinonchat, 2022) proposed an algorithm that recognize the load signatures extracted from the kurtogram technique and works well for high power consumption loads. Likewise, proposals have been developed that unite the use of MLP together with other statistical models,

such as goodness of fit (A. R. Rababaah & E. Tebekaemi, 2012), to detect and classify events produced by residential loads.

- **Recurrent Neural Network**

Recurrent Neural Networks (RNNs) are feedforward neural networks that are extended and include feedback connections. These ANNs are commonly uses to process sequential data or time series data such as in language translation or speech recognition applications. They are distinguished by their memory, so the assumption that the inputs and outputs are independent of each other is no longer valid. However, these architectures tend to run into the vanish gradient problem. This problem is that the size of the gradient, which is defined as the slope of the loss function along the error curve, continuously decreases by updating the weight parameters until they become insignificant and therefore the algorithm can no longer learn. Short-Term Long Memory units (LSTMs) (S. Hochreiter & J. Schmidhuber, 1997) try to overcome this problem by using their forget gate that allows to decide whether or not the information should be forgotten and update the parameters of the model accordingly. In addition to the LSTM architectures there are other variants of RNNs, for instance, the Bidirectional Recurrent Neural Networks (BRNNs) (M. Schuster & K. Paliwal, 1997) and the Gated Recurrent Units (GRUs) (K. Cho et al., 2014). In relation with this last variant, in (T. Le et al., 2016) it is implemented as an energy disaggregation classifier with results comparable to previous work. More recently, in (H. Rafiq et al., 2020) a disaggregation algorithm based on a deep LSTM network has been proposed for low-frequency electrical measurements.

- **Convolutional Neural Network**

Convolutional Neural Networks (CNNs) (K. O'Shea & R. Nash, 2015) are similar to feedforward networks, but these algorithms use the principles of linear algebra, particularly matrix multiplication, to identify patterns within an image. For that reason, they are the most applied to analyse images in tasks such as image classification and object recognition. The layers of these architectures are arranged in such a way that the neurons cover the entire visual field of the input data, avoiding the problem of fragmentary processing of other neural networks. Hidden layers include multiple convolutional layers, pooling layers, fully connected layers, and normalization layers; which are designed to reduce processing requirements throughout the architecture. With regard to the application of these networks in NILM, for example in (F. Ciancetta et al., 2021) the proposal recognizes a device whether it works in a singular way or in combination with other loads with low error rates for event detection and classification. Furthermore, (H. Grover et al., 2022) has proposed an architecture called Mh-Net CNN to identify the energy consumption and time-of-use of individual appliances connected in a building microgrid.

- **Autoencoder**

Autoencoders are the benchmark unsupervised neural network (P. Baldi, 2011). They are defined as feedforward networks that aim to regenerate the input from the reduced dimensionality encoding by ignoring insignificant data, such as noise. Within this type of ANN there are variants with the aim of changing the useful properties that must be learned in the training process. For instance, regularized autoencoders (Y. Bengio et al., 2013) are effective in learning representations for subsequent classification tasks, and variational autoencoders (D. P. Kingma & M. Welling, 2019), with applications as generative models. In (S. Verma et al., 2021), where a LSTM autoencoder is proposed, NILM is presented for the first time as a dynamic modelling problem, while it is posed as a multi-label classification. Moreover, autoencoders used in NILM usually treat unconcerned devices as “noise”, such as in (X. He et al., 2022) where the proposed method is based on a denoising autoencoder, which is able to identify the consumption and the operation state.

The main advantage of ANN approaches over other statistical methods is that they are non-parametric models, this implies that they need less statistical background and can therefore be seen as simpler algorithms to use. Nevertheless, they have some disadvantages related to the computational cost which is considerable and implies long training periods, reaching even days, and the requirement to use a large amount of data also in this process. Additionally, ANNs are black box models that do not have the ability to interpret the relationship between inputs and outputs, but also, they cannot deal with uncertainties in the parameters of the model or in the observed data. Even so, it should be noted that ANNs represent a useful tool for processing data, since they have great capacity to extract the relevant information that allows to derive the hierarchy of the characteristics and thus be able to perform the tasks for which they were specifically created by training the architectures beforehand.

2.3 Summary

The current chapter has presented the available knowledge of NILM techniques, and the methods used for the load disaggregation stage of its procedure. The main objective is to offer the most relevant information on the subject on which this project is based to facilitate the understanding of the subsequent chapters and the reason why this thesis has been developed. For that purpose, an explanation of what the term NILM means has been developed, the set of actions involved in these techniques has been detailed, and their main applications have been presented. In addition, a review of the algorithms used for the distinction of household loads has been included, distinguishing between supervised and unsupervised learning approaches. On the other hand, the ANNs have been examined in more depth in a separate section, due to their relevance. In this part of the chapter, different topologies have been analysed to illustrate the variety of existing architectures depending on the applications. Table 1 summarizes the provided information on some of the most recent NILM-related work that used the methods discussed above. Finally, the chapter continues to present the advantages and disadvantages of ANNs compared to other methods, justifying their choice for the development of this Master's Thesis.

Table 1. Summary of the methods proposed in previous works.

PREVIOUS WORKS	METHOD
(T. -T. -H. Le et al., 2020)	FFT-BDT
(M. Nguyen et al., 2015)	DT
(A. S. Hernández et al., 2021)	SVM
(Y. Lin & M. Tsai, 2011)	HSVM
(C. C. Yang et al., 2017)	Naïve Bayes Classifier
(P. Adjei et al., 2020)	Gaussian Naïve Bayes Classifier
(B. Bertalanic et al., 2021)	SVM, kNN, MLP, LR, RF
(C. C. Yang et al., 2018)	kNN and Naïve Bayes Classifier
(Z. Xiao et al., 2021)	RF (cooling loads)
(Wu et al., 2019)	RF
(H. Kim, 2011)	FHMM, FHSMM, CFHMM and CFHSMM
(A. Moradzadeh et al., 2020)	PCA
(S. Qi et al., 2021)	PCA
(P. G. Papageorgiou et al., 2021)	DFFT and k-Means
(A. Yasin & S. A. Khan, 2018)	Event detection, k-Means, and on-off paring
(S. Yaemprayoon & J. Srinonchat, 2022)	MLP
(A. R. Rababaah & E. Tebekaemi, 2012)	MLP and goodness of fit
(T. Le et al., 2016)	GRU network
(H. Rafiq et al., 2020)	Deep LSTM network
(F. Ciancetta et al., 2021)	CNN
(H. Grover et al., 2022)	Mh-net CNN
(S. Verma et al., 2021)	LSTM autoencoder
(X. He et al., 2022)	Denoising autoencoder

Chapter 3 Proposed architectures

This chapter will describe the proposed architectures for the identification of the loads. Altogether, three different architectures have been defined: the first architecture is based on RNN and the other two, on CNN. However, all of them share the same processing structure, which is designed in such a way that there is a first stage, where the input samples are adapted for the proper function of the second and last stage, which deal with the classification of the load making use of neural networks.

3.1 Recurrent architecture

The following sections will explain in detail the pre-processing and load classification stage for the recurrent topology approach. It is worth mentioning that this first approach is based on the one presented for a low sampling frequency analysis in (L. de Diego Otón, 2020).

3.1.1 Pre-processing Stage

In this architecture, the measured raw electric current signal is used as a source of information about the use of the different household appliances by examining the effects produced in this magnitude. Although this signal is initially sampled at a frequency of 12kHz, to decrease the amount of data to be processed, this frequency is reduced to 4kHz by taking one out of every three samples. As a consequence, it is true that this process involves the deterioration of the accuracy of the captured waveform, as can be observed in Figure 5. From an analysis of the frequency components, some information is lost at frequencies above 4 kHz, although this information cannot be seen in the figure. Additionally, this process can be considered as an adaptation to the operation of commercial Analog Front-Ends (AFE), such as the one used for the experimental tests carried out in the context of the project to which this proposal belongs.

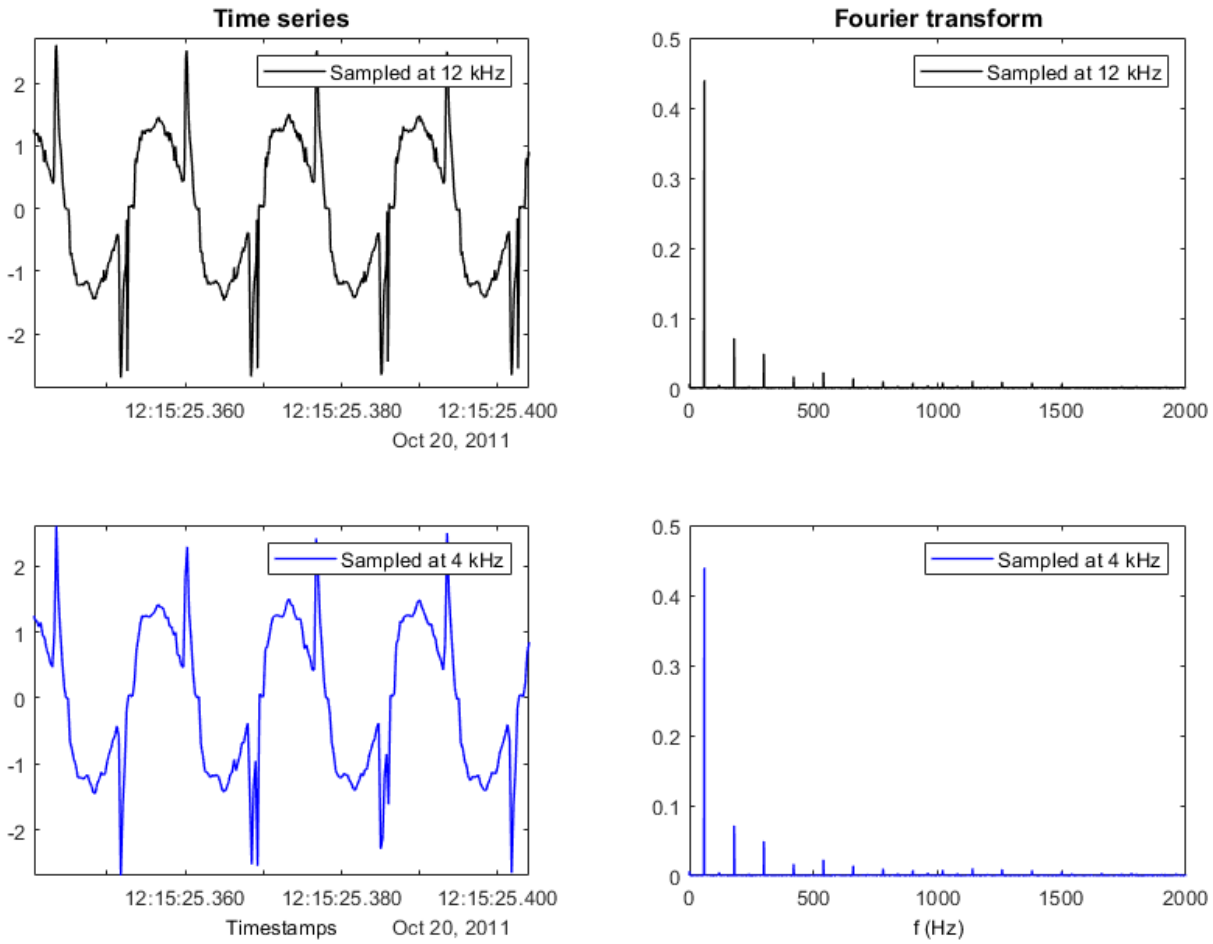


Figure 5. Effects in the time domain and frequency components analysis.

The inputs for the following stage are temporal windows around the detected events captured from the electric signal. In practice, these events are commonly defined as state transitions which exceeds a certain consumption level and has a duration longer than a defined minimum interval. Moreover, this is also a way to detect them through an intuitive processing which considers the amplitude and length of these noticeable changes in the signal. It is noteworthy that, by doing so, noise and oscillations introduced into the signal due to the own operation of the electrical devices or the measuring instruments are filtered. On the other hand, and as stated above, among the electronic devices there are different types of operating states, but also, consumption levels vary. Hence, the effects on the current of low-power devices may be less appreciable compared to those with higher consumption at the classification stage. For that reason, and in order to use a common scale, the signal is normalized by adjusting the values between -1 and +1 within each window. This specific range is selected due to the fact that the units of the classification architecture use functions for the processing of these inputs that saturate at those limits and, therefore, vanishing gradient problems may appear if they are exceeded. Besides the limits on the amplitude of the signal, another important aspect that will characterize the input windows for the classification stage is the length. This length should be defined in such a way that the window provides enough information on the changes that occurred at the change of state and during its subsequent transient. Experimentally, its value has been set at 4096 samples. This length for a 4 kHz frequency constitutes a time period of 1.024 s, which contains between 51 or 62 cycles of the electric current signal based on the fact that the fundamental frequency of the power grid is 50 Hz or 60 Hz, depending on the country. Regards to the above, the resulting temporary windows are seen as the examples shown below in Figure 6. The same events will be used for the rest of the architecture examples.

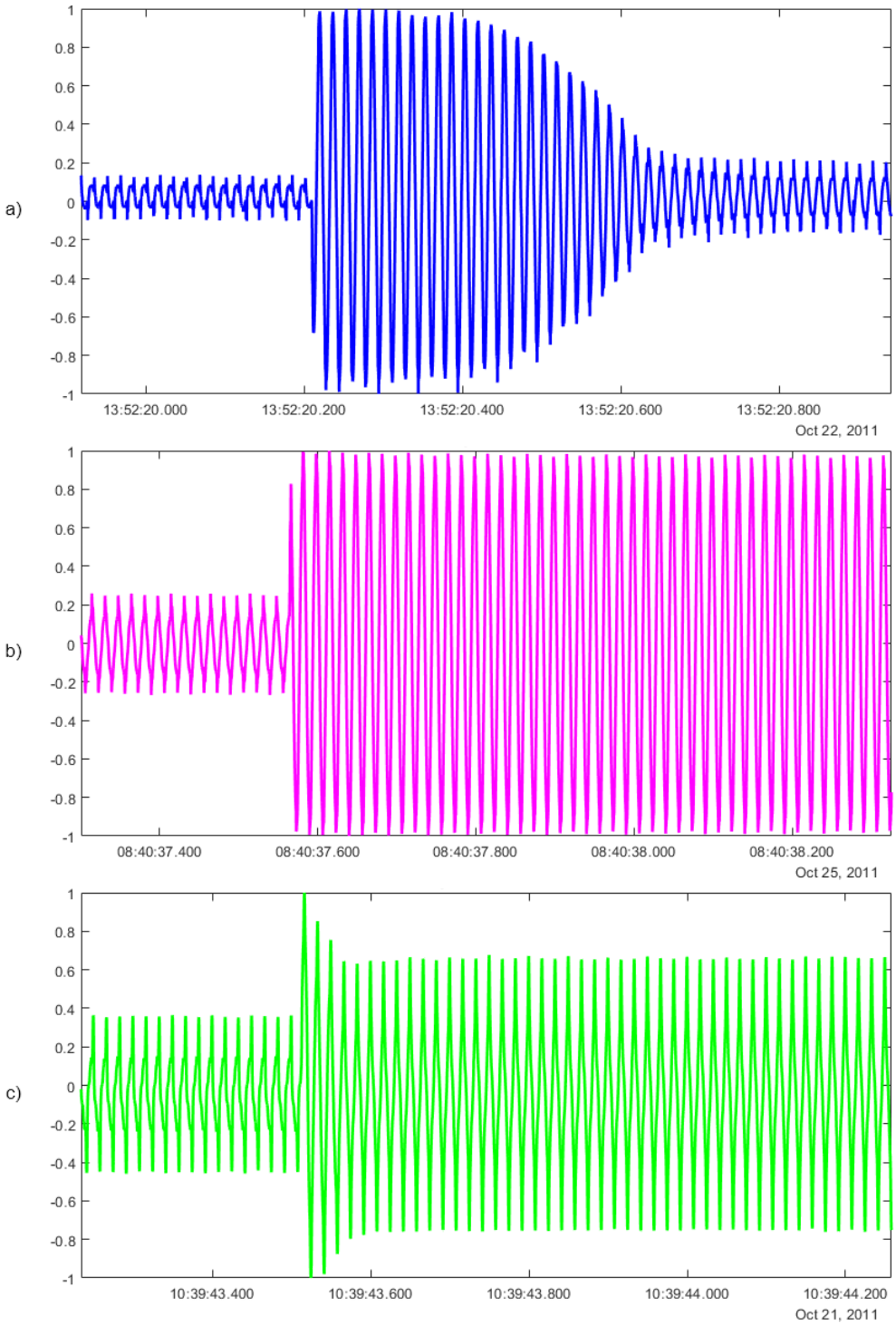


Figure 6. Examples of input samples for the RNN of on events of (a) refrigerator, (b) hair dryer and (c) garage door.

In accordance with this section, one point that should be underlined is that, although the classification system does not have information about the actual amplitude of the detected events, the waveform of the signal will become the fundamental factor in distinguishing electrical loads.

3.1.2 Load Classification Stage

In this second stage, the goal is to classify events within the current according to the device that produced them. For that proposed, in this first approach a recurrent topology based on LSTM cells will be used. Hereafter, a brief outline of the structure of these cells is included in order to properly understand how they are able to learn long-term dependencies over time.

They are composed of a cell state c_t which is capable of remembering values from arbitrary previous time intervals; a hidden state h_t also considered as the output vector of the LSTM unit; and three gates that control the data flow that enters, exits, and remains within the cell, respectively: input gate i_t, output gate o_t, and forget gate f_t. In addition, there is an extra cell called candidate g_t that could be added to the state or not. The structure of the LSTM units is shown in Figure 7 and its operation follows the steps below:

1) Firstly, the information to discard is selected (3.1). This decision is made by the forget gate f_t which takes the previous hidden state h_{t-1}, together with the current input x_t, and outputs a number between 0 and 1 using a sigmoid function σ (3.7). The meaning of this output oscillates between getting rid of this information and keeping it, respectively.

$$f_t = \sigma(W_f \cdot x_t + R_f \cdot h_{t-1} + b_f) \tag{3.1}$$

2) The following step is to decide the new data that is going to be stored in the cell state c_t where two contributions are combined. On the one hand, another sigmoid function that makes the choice of which values will be updated in the input gate i_t (3.2). On the other hand, a hyperbolic tangent function $tanh$ (3.8) in the candidate cell g_t has the capacity of adding new values to the state (3.3).

$$i_t = \sigma(W_i \cdot x_t + R_i \cdot h_{t-1} + b_i) \tag{3.2}$$

$$g_t = tanh\big(W_g \cdot x_t + R_g \cdot h_{t-1} + b_g\big) \tag{3.3}$$

3) Once the forget gate and the input gate are calculated, it is time to update the previous cell state c_t (3.4). In short, the values to be forgotten from the old state are removed and the new candidates are added, scaled by how much it is decided to update them.

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{3.4}$$

4) Finally, the output of the cell remains to be calculated (3.5). The cell state c_t is passed through a hyperbolic tangent function to scale its values between -1 and 1 and them, filtered using the output gate o_t (3.6) which is obtained by a sigmoid function.

$$h_t = o_t \odot tanh(c_t) \tag{3.5}$$

$$o_t = \sigma(W_o \cdot x_t + R_o \cdot h_{t-1} + b_o) \tag{3.6}$$

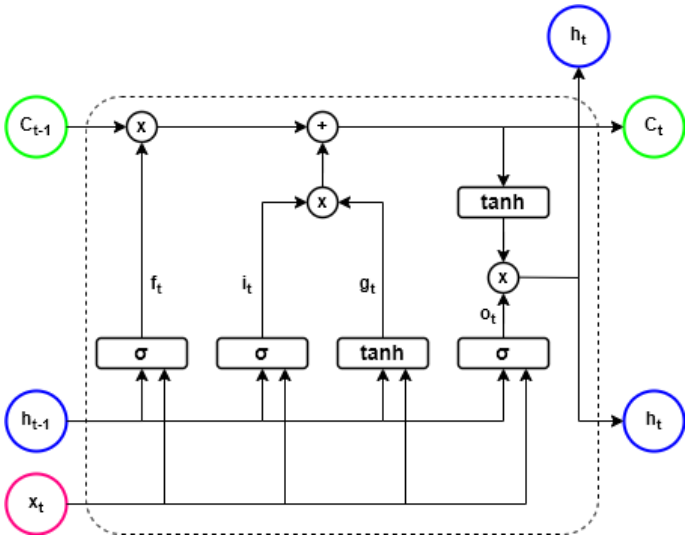


Figure 7. Flow data diagram of the LSTM cell.

It is important to make clear some aspects mentioned above. For instance, the symbol ⊙ represents the Hadamard product (G. P. H. Styan, 1973) which is a matrix binary operation, similar to matrix addition, that produces an array of the same dimensions as the two matrixes of operands and where the elements are the product of the elements of the original matrices corresponding to the same row and column. Moreover, the sigmoid (3.7) and hyperbolic tangent (3.8) are activation functions (J. Lederer, 2021). As has been observed, this type of function in the context of neural networks defines how the weighted sum of the input is transformed into an output, incorporating the nonlinear behaviour of the cell. Both functions have a characteristic S-shape, as can be observed in Figure 8 and Figure 9. Particularly, the hyperbolic tangent function is shifted and scaled version of the sigmoid or logistic function.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.7}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.8}$$

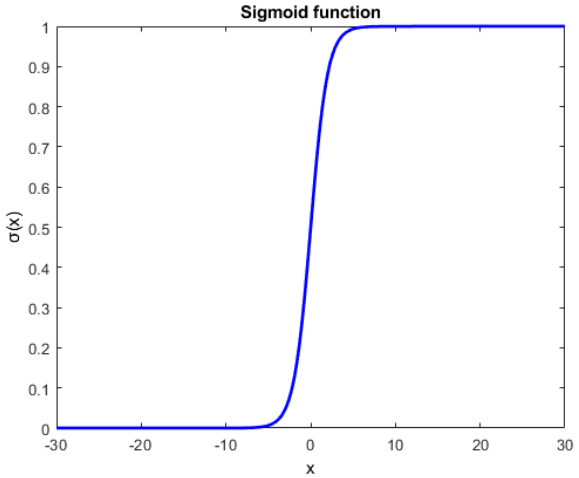


Figure 8. Representation of the sigmoid function (3.7).

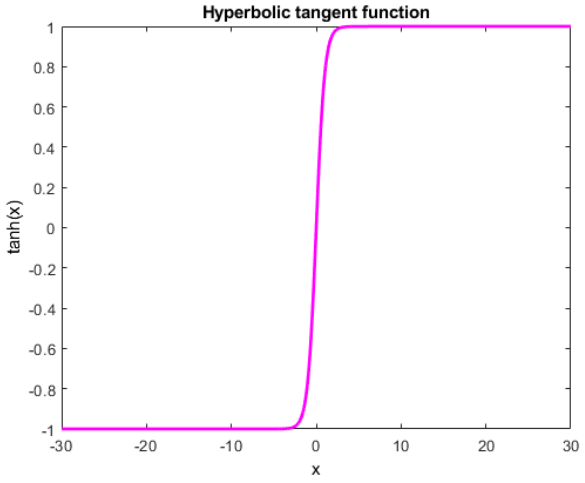


Figure 9. Representation of the hyperbolic tangent function (3.8).

In addition to the LSTM layer, that has been described in detail due to its relevance for the functionality of this specific network, the rest of the layers will be explained hereunder together with the structure, shown in Figure 10. Specifically, the architecture used in this first approach is similar to the first version defined in (L. de Diego Otón, 2020).

- **Input Layer**

It is used as an entry point to the network. Within its parameters, the input shape matches the dimensions of the temporal windows.

- **LSTM 1 Layer**

This first LSTM, as well as the next, will extract the relevant information throughout the input sequence. In this case, the number of units corresponds to 1/20 of the length of the input sequence because after a set of tests this configuration produced the most outstanding results. In addition, the activation function by default are the ones previously described. On the other hand, another key parameter of this specific layer is that it has been defined so that it provides the full sequence output rather than the last output value to the LSTM layer below. This is in order to build a stacked LSTM configuration that adds depth in space to the inherent temporal depth of these types of layers. This structure was first used in (A. Graves et al., 2013) for its application in speech recognition.

- **LSTM 2 Layer**

As mentioned, this layer works alongside the previous one because, in addition to being able to create a more complex feature representation of the current input, increasing the depth of the network provides an alternative solution that requires fewer neurons and trains faster (A. Graves et al., 2013). Moreover, with respect to the previous LSTM layer, the number of units is halved to reduce the information used for classification.

- **Dense Layer**

It performs the classification process, so the number of units, and consequently the number of outputs, corresponds to the number of devices involved in the energy disaggregation. From this layer it stands out that the activation function in this case is the one called as SoftMax or normalized exponential function (J. Lederer, 2021). This is because a function that converts the information extracted from the stacked LSTM

layers into a probability distribution is needed for the sorting. This function is defined by the formula (3.9) where it is applied standard exponential function to each element x_i of the vector $\boldsymbol{x}$ and normalizes these values by dividing it by the sum of all the exponentials. This calculation ensures that the sum of the output is 1.

$$\sigma(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \quad for\ i = 1, \dots, K\ and\ \boldsymbol{x} = (x_1, \dots, x_K) \in \mathbb{R}^K \tag{3.9}$$

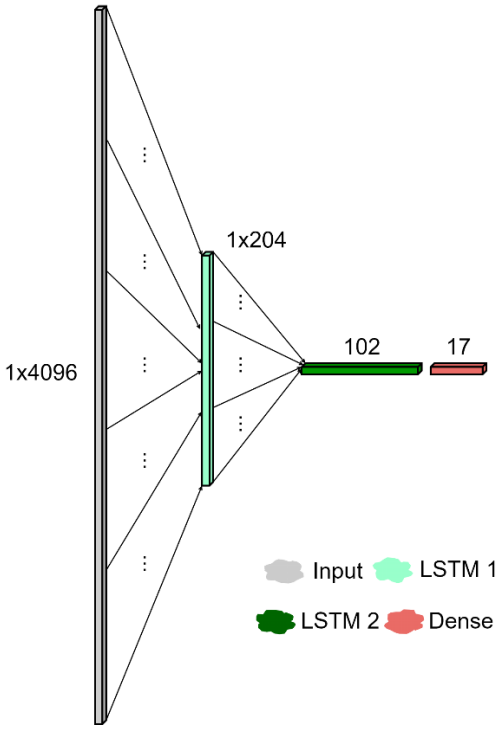


Figure 10. Structure of the proposed recurrent neural network (L. de Diego-Otón et al., 2022).

3.2 Convolutional architectures

The section will describe both stages of the two CNN-based structures differentiated by the relevant characteristics extracted from the input samples in the time or frequency domain, respectively. In addition, it should be noted that the one that works in the time domain was presented in (L. de Diego-Otón et al., 2021).

3.2.1 Pre-processing Stage

Temporary windows used as inputs samples of the network from the first proposed architecture will also be used for these architectures. Nevertheless, since the neural network topology used in these approaches is convolutional, and they are commonly used to work with images, the temporary samples will be transformed into images.

For the first CNN-based architecture, the temporal window is divided into sections with a fixed length, these sections are introduced into the rows of a matrix, and, eventually, this matrix is transformed into a grayscale image. The image creation process was presented in (L. de Diego-Otón et al., 2021). Since the total length of the temporal window is 4096, the time window is divided into 64 sections with a length of 64 samples, in order to obtain squared images with dimensions of 64×64. Regarding this matter, the possibility

of using other lengths for the time window had been analysed, however, smaller dimensions imply not having enough information about the transient state, whereas higher dimensions require an unnecessary computation load because most of the window does not have relevant information. Furthermore, the possibility of use images created with overlaps between sections had also been studied to focus the attention on the event, downplaying the fragments before and after the event. Nonetheless, images without applying any overlap have only been considered in order to take these permanent regimes into account in the classification of loads and due to the fact that, in general, although non-overlapping can result in event losses if the change is between two divisions, the performance of classifiers is comparable to that obtained with overlapping (A. Dehghani et al., 2019). Figure 11 shows an example of the resulting images used for this first architecture.

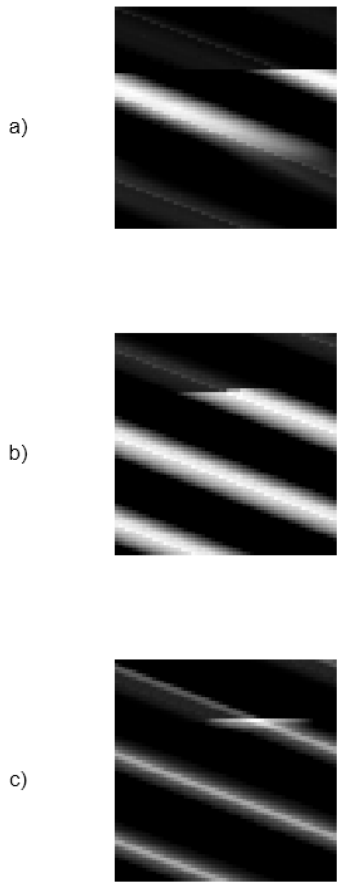


Figure 11. Examples of input samples for the first CNN of on events of (a) refrigerator, (b) hair dryer and (c) garage door.

In the second architecture, the input images of the classification stage are obtained from the outcomes of the Short Time Fourier Transform (STFT) of the temporal windows (K. Gröchenig, 2001). This process consists of a Fourier-related transformation of local sections, which is performed to determine the frequency and phase content of the signal as it changes over time. It should be noted that whereas the STFT is used in conditions where the frequency components vary over time, the standard FTT provides the averaged information for the entire time interval of the signal (N. Kehtarnavaz, 2008). Moreover, STFT results are 2D matrices made up of complex numbers where the information is represented as frequency versus time, often represented as a graph called Spectrogram. However, when transforming these matrices into images it is sought not to lose information, so these complex numbers are divided creating two different matrices. With regard to this division, two different forms of division have been considered: distinguishing

between magnitude and phase, or between the real and imaginary part. The values of the matrices obtained are expressed in grayscale and divided into two because they have an absolute symmetry. For that reason, the resulting dimension is 320×320 and the final input images used for the classification are such as those shown in Figure 12.

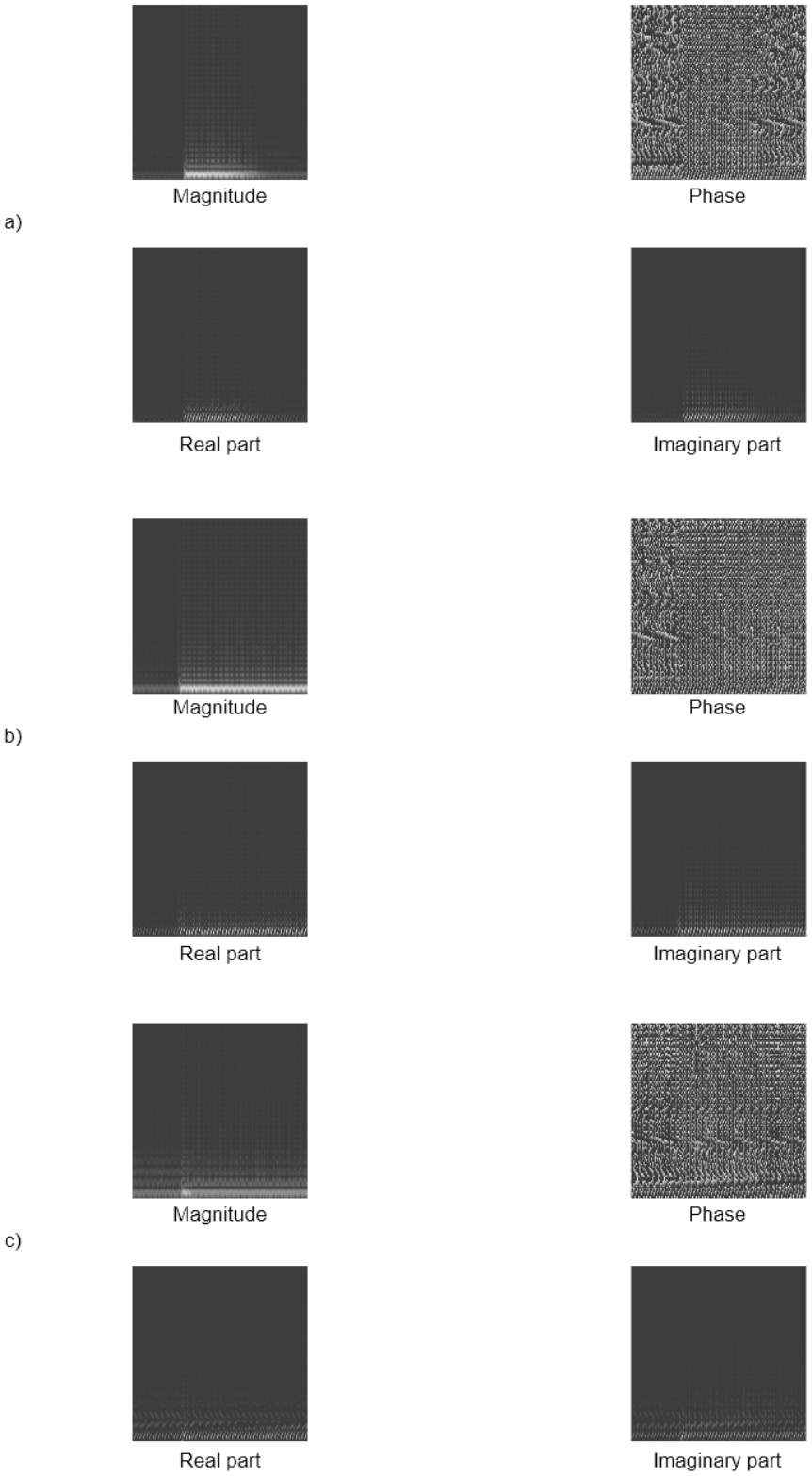


Figure 12. Examples of input samples for the second CNN of on events of (a) refrigerator, (b) hair dryer and (c) garage door.

3.2.2 Load Classification Stage

For the classification of the images created in the previous stage, as has already been mentioned, a convolutional neural network topology will be used. The core component used in this type of algorithms is the convolutional layer, as its name suggests. For that reason, it will be the one to focus on before entire setup is described.

As said, the convolutional layer performs an operation that gives it its name, the convolution. This mathematical operation performed on two functions (typically f and g) to obtain a third function. Explicitly, it is defined as the integral of the product of the two functions, having one of them reversed and shifted (3.10). The convolution operation is typically denoted with an asterisk.

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau) \cdot g(t - \tau)\, d\tau \tag{3.10}$$

In the context of neural networks or image processing, the convolutional operation involves the multiplication of an array of input data with a filter or kernel, which is defined as a two-dimensional array of weights, obtaining a feature map. The process is represented in Figure 13. However, some convolutional neural networks in machine learning libraries are actually implemented using cross-correlation (P. Bourke, 1996) instead of convolution. The main difference is that in convolutional operation the kernel is flipped before sliding it through the image, whereas in cross-correlation the filter is used directly. The only reason to flip the kernel is to get the commutative property, which is usually not of great relevance in the implementation of a neural network because this does not change the model's performance, but weights are just learned flipped and, therefore, cross-correlation is used. The size and values of the filter determine the transformation effect of the convolution process. Making the kernel smaller than the input, the network has sparse interaction which means that the memory requirements are reduced. Another important aspect related to storage requirements is that each kernel member is used at each position of the input, so instead of learning a separate set of parameters for each location, only one set is learned. In addition, this specific form of parameter sharing causes that the convolutional layer also has the property called equivariance, in other words, the output change in the same way as the input (I. Goodfellow et al., 2016).

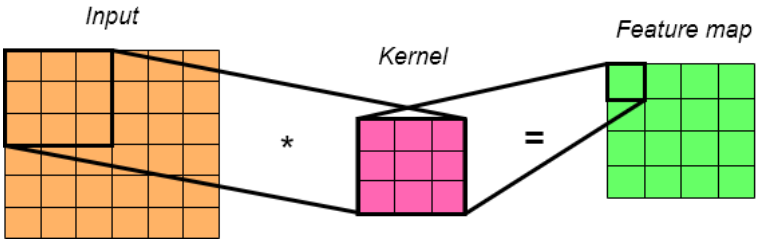


Figure 13. Convolutional operation graphical representation.

As in the recurrent case, the configuration of the neural network, shown in Figure 15, and the rest of the layers will be defined thereupon. However, since there are two approaches to the pre-processing stage, two different neural networks were also created. For the first configuration that works in the time domain, the layers that form it are:

- **Input Layer**

It is used to introduce the images into the neural network and its shape corresponds to the images' dimensions, which is 64×64.

- **Convolutional 2D 1 Layer**

This first convolutional layer, as well as the next one, processes the images in order to extract the main characteristics of the localized events. Within the configuration, the kernel used is of 3×3 and the output space dimensionality is 16, since this is the number of filters that are used in this layer.

- **Max Pooling 1 Layer**

After each convolutional layer, there will be a pooling layer that will operate on top of the feature maps to create new smaller ones. As a result, this process also makes it possible to reduce the subsequent layers. The window size is 2×2 which means that each dimension is halved. In addition, this process involves the selection of the pooling operation and, as the name suggests, in this case, it will calculate the maximum value for each patch of the feature map.

- **Convolutional 2D 2 Layer**

As can be derived, there is more than one convolutional layer not only to improve the ability of extracting features, but also to obtain more complex characteristics. In this case, the dimensionality of the output space is 32, with again a kernel size of 3×3.

- **Max Pooling 2 Layer**

As mentioned, this layer is used to reduce the size of each feature map, also in this case, by a factor of 2 because the size of the pooling filter is 2×2, prevailing the relevant characteristics of the images.

- **Flatten Layer**

This layer is used to transform the multidimensional input into a one-dimensional output whose shape is equal to the total number of elements contained in the feature map. Commonly used in the transition from the convolutional layer to the full connected layer, as on this occasion.

- **Fully connected Layer**

The flattened high-level features extracted from the previous layers are fed into the Fully Connected layer. Within this layer, each neuron has a connection with each of the next sub-layer, that is, there is a flow of information between each element of the feature map and this layer is able to extract the relation between all of them. Regarding the design of this layer, the number of units corresponds to the number of devices used in the classification process, and the activation function is the Rectified Linear Unit (ReLU). This function is defined as the positive part of its argument (3.11) and represented by a ramp whose slope starts in zero, as shown in Figure 14 (A. F. Agarap, 2018).

$$f(x) = \max(0, x) = x^{+} \tag{3.11}$$

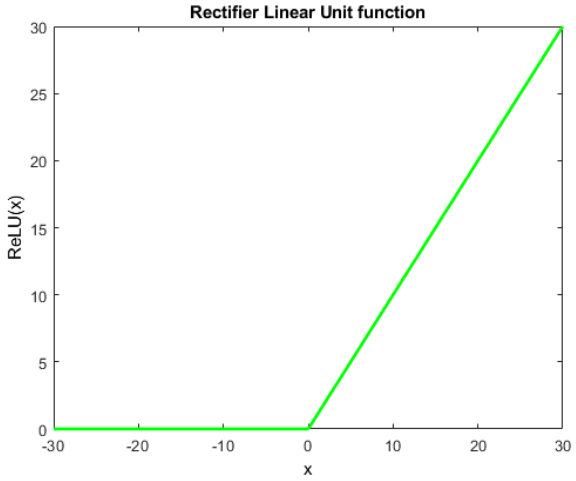


Figure 14. Representation of the Rectified Linear Unit function (3.11).

- **Dense Layer**

As in the recurrent architecture, the last layer that is responsible for classifying events is a dense one whose activation function is the SoftMax.

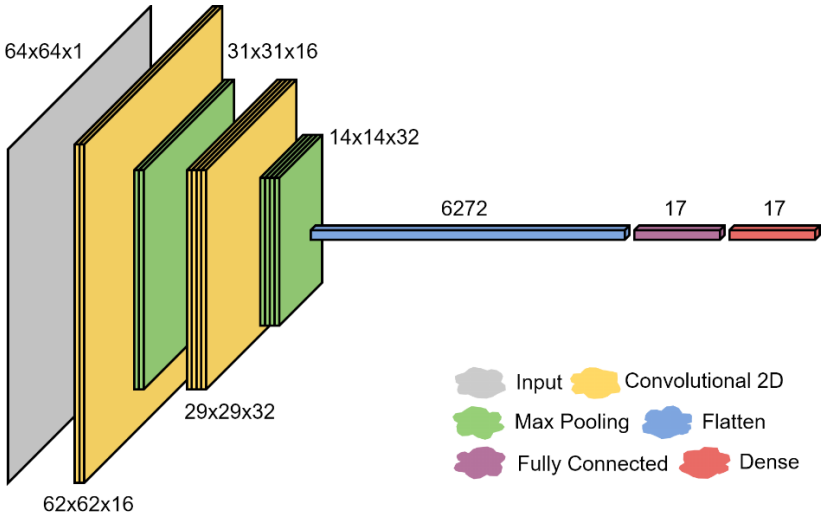


Figure 15. Structure of the convolution neural network proposed for time domain inputs (L. de Diego-Otón et al., 2022).

On the other hand, and as can be observed in Figure 16, the second configuration is made up by two branches based on the previous structure but with three stages for the characteristic extraction. The reason for including an extra stage is that the input images are much larger than in the previous case. Moreover, a concatenate layer is included in order to used jointly the processed information obtain in each branch from each pair of images. Be that as it may, the layers forming the neural network are:

- **Input 1 Layer and Input 2 Layer**

The input size corresponds to 320×320 which is the size of the STFT images.

- **Convolutional 2D 1 Layer**

The kernel used in this architecture has the same dimension as in the previous case, this is to say, 3×3 in all the convolutional layers. Additionally, the number of filters for this first layer is 16, and through the successive layers this parameter will increase by 2.

- **Max Pooling 1 Layer**

The size of the filter will be 2×2 in this layer and the next two, because the goal is to divide in half the size of each feature map.

- **Convolutional 2D 2 Layer**

As mentioned earlier, the filter number is twice that of the first convolutional layer, which is the same as an output spatial dimensionality of 32.

- **Max Pooling 2 Layer**

As stated above, the dimensions of the filter are 2×2.

- **Convolutional 2D 3 Layer**

In the same way as before, in this layer the dimensionality of the output space increases to double with a number of filters of 64.

- **Max Pooling 3 Layer**

Following the same definition, 2×2 is the filter size in this layer.

- **Concatenate Layer**

This layer, as indicated above, is used to connect both branches to use the relevant characteristics of both images together in the final layers.

- **Flatten Layer**

It resizes the output dimension of the previous layer so that this information can be entered into the Fully Connected layer.

- **Fully connected Layer**

This layer has the capacity of extracting the relation between the elements of the features maps flattened through the interconnections of its units. In this case, the number of units does not correspond to the number of classes of devices used in the classification process because the trials indicate that it was a value too low for the transition. For that reason, it was defined for the value squared, 256.

- **Dense Layer**

Classification is performed on this layer using the SoftMax activation function. One aspect that may catch the reader's attention is that, in this case, the dimension is not 17, but is 16. This is because for classification one of the device classes is discarded, as will be explained in detail in the next chapter.

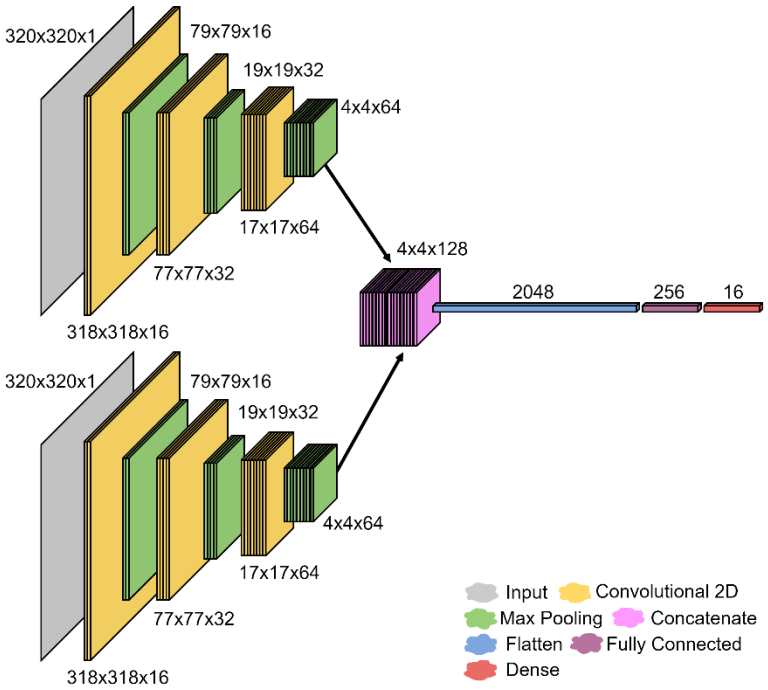


Figure 16. Structure of the convolution neural network proposed for frequency domain inputs (L. de Diego-Otón et al., 2022).

3.3 Summary

For the load identification stage, three different solutions have been presented. All of them use processed sections around the on/off events of the current signal, sampled at high frequency, as input to the neural networks that are included in the architectures. From these portions of the signal the information of interest of the time domain was extracted, except in the last case that uses the frequency domain features. Regarding the type of neural network, the first architecture contains a recurrent one based on LSTM layers, whereas for the other two solutions the topology is the convolutional one. Furthermore, all the layers functionalities and configurations were commented in detail.

Chapter 4 Experimental results

In this chapter, the main findings will be presented and interpreted in order to show its significance to the research objectives. In addition, there will be an introductory section where contextual aspects will be addressed to better understand the achievement of these results.

4.1 Context

Validation of the algorithms discussed above will be performed using residential electricity usage samples provided by public databases due to the time-consuming process of making a real-world data collection. Table 2 compares the most renowned databases in terms of NILM. It is worth mentioning that, in this field, residential measurements are dominant against industrial datasets.

The dataset used for this project is BLUED (*"Building-Level fUlly labelled Electricity Disaggregation"*). Among the great variety of this type of publicly available databases, this one has been chosen specifically due to its characteristics. The measured variables are the aggregated current-voltage pair sampled at a frequency of 12 kHz, considered high frequency, obtained from a single-family residence in Pittsburgh, Pennsylvania, U.S. Furthermore, the measurements were performed for a week. Based on aggregated captured signals, active and reactive power have been calculated at a rate of 60 Hz and included in this dataset. In addition, ground-truth is provided by labelling and time-stamping each state transition, or events, of the loads in the house thanks to plug-level meters, environmental sensors, and circuit panel meters. In this particular dataset, events are defined as changes of more than 30 Watts in power consumption that last, at least, 5 seconds.

As discussed at the beginning of Chapter 3, inputs for the classification stage in all approaches were based on temporal windows around detected events. The exact location of events through the captured signal is carried out using the provided ground-truth, taking into account the down-sampling process implemented at the beginning of the pre-processing stage. Within the temporal windows the timestamp of each event will be located in the first quartet since it is necessary to include also its subsequent transient.

Regarding the appliances, Table 3 shows the list of the specific electrical devices monitored during the data acquisition stage and the reorganization carried out to use them for the experimental testing of the approaches. In short, there are seventeen classes whose labels are those in the right column: lights and lamps of all rooms in the house (00), subcircuits of the distribution panel (01), garage door (02), kitchen aid chopper (03), refrigerator (04), A/V system of the living room (05), computer 1 (06), laptop 1 (07), basement receiver/DVR/Blu-ray Player (08), air compressor (09) LCD monitor 1 (10), television (11), printer (12), hair dryer (13), iron (14), empty socket of the living room (15), and monitor 2 (16). The total number of events per class obtained from the database is presented in Table 4.

It is worth mentioning that labels 00 and 01 involve the events of different electrical devices, as can be observed in Table 3. In the case of lights and lamps, they are categorized together in this way because their consumption can be considered similar and, when distinguishing these loads separately, their classification can become more complex. On the other hand, devices that were not easily measured with plug meters or

environmental sensors were included in the database through the five distribution circuits. These subcircuits are all together because they represent a specific part of the electrical infrastructure, and due to the fact that the devices they comprise are not known. An important aspect to mention is that this lack of knowledge may negatively influence on the results of these approaches due to their supervised nature; however, a future unsupervised solution could solve this problem by grouping each event separately.

Table 2. Comparison between NILM Datasets.

NAME	CITATION	DATE	COUNTRY	SAMPLING FREQUENCY	DURATION	GROUND TRUTH
ACS-Fl	(C. Gisler et al., 2013)	-	Switzerland	10 seconds	2 sessions of 1 hour each	Plug-meters
AMPds	(S. Makonin et al., 2016)	2015	Canada	1 minute	2 years	Submeter channels
BLOND	(T. Kriechbaumer & H. Jacobsen, 2018)	2018	Germany	50 kHz to 250 kHz	50 to 213 days	Individual appliances
BLUED	(K. D. Anderson et al., 2012)	2012	US	12 kHz /60 Hz	8 days	Labelled events
COMBED	(N. Batra et al., 2014)	-	India	30 seconds	1 month	-
COOLL	(T. Picon et al., 2016)	2016	France	100 kHz	840 waveforms (of 6 s each)	Individual appliances
Dataport	(O. Parson et al., 2015)	2013	US	1 Hz to 1 minute	4 years	Submeter channels
ECO	(C. Beckel et al., 2014)	-	Switzerland	1 second	8 months	Submeter channels
GREEND	(A. Monacchi et al., 2014)	2014	Austria, Italy	1 Hz	3-6 months	-
HES	(J. P. Zimmermann et al., 2012)	2012	UK	2 minutes	1 year /1 month	Submeter channels
HFED	(M. Gulati et al., 2014)	2015	-	10 kHz to 5 MHz	-	-
iAWE	(N. Batra et al., 2013)	2013	India	1 Hz	73 days	Submeter channels
IHEPCDS	(K. Bache & M. Lichman, 2013)	-	France	1 minute	4 years	-
LIT	(D. P. Renaux et al., 2020)	-	Brazil	15 kHz	30s to hours	Individual appliances + labelled events
PLAID	(J. Gao et al., 2014)	2014	US	30 kHz	1094 waveforms (of 1 s each)	Individual appliances
RAE	(S. Makonin et al., 2018)	2018	Canada	1 Hz	72 days	Submeter channels
RBSA	(B. Larson et al., 2014)	-	Pacific Northwest, USA	15 minutes	27 months	Submeter channels
REDD	(J. Kolter & M. Johnson, 2011)	2011	US	16,5 kHz /1 Hz	119 days	Submeter channels
REFIT	(D. Murray et al., 2017)	2017	UK	8 seconds	2 years	Submeter channels
Smart*	(J. Schultz et al., 2000)	2012	US	1 Hz	3 months	-
SustDataED	(M. Ribeiro et al., 2016)	2016	Portugal	12.8 kHz	10 days	Individual appliances
SynD	(C. Klemenjak et al., 2020)	2020	Austria	5 Hz	180 days	-
Tracebase	(A. Reinhardt et al., 2012)	2014	Germany	1 Hz	1 day	Individual appliances
UK-DALE	(J. Kelly & W. Knottenbelt, 2015)	2015	UK	16 kHz /1 Hz	2 years	Submeter channels
WHITED	(M. Kahl et al., 2016)	2016	Multiple	44 kHz	5123 waveforms (of 5 s each)	Individual appliances

Table 3. Electrical devices monitored.

NAME	LABEL	
Living Room Desk Lamp	00	(1)
Living Room Tall Desk Lamp	00	
Garage Door	02	
Kitchen Music	-	
Kitchen Aid Chopper	03	
Refrigerator	04	
Living Room, A/V System	05	
Sub-Woofer LR	-	
Computer 1	06	
Laptop 1	07	
Basement Receiver/DVR/Blu-ray Player	08	
Sub-Woofer Basement	-	
Air Compressor	09	
LCD Monitor 1	10	
TV	11	
Hard Drive	-	
Printer	12	
Hair Dryer	13	
Iron	14	
Living Room Empty Socket	15	
Monitor 2	16	
Backyard Lights	00	
Washroom Lights	00	
Office Lights	00	
Closet Lights	00	
Upstairs hallway Lights	00	
Hallway stairs Lights	00	
Kitchen hallway Lights	00	
Kitchen overhead Lights	00	
Bathroom upstairs Lights	00	
Dining Room overhead Lights	00	
Bedroom Lights	00	
Basement Lights	00	
Circuit 4	01	(2)
Circuit 7	01	
Circuit 9	01	
Circuit 10	01	
Circuit 11	01	

(1) Class 00 called as “Lamps and Lights”.

(2) Class 01 called as “Distribution Circuits”.

Table 4. Distribution of database events per classes.

LABEL	CLASS NAME	NUMBER OF EVENTS
00	Lights and lamps	1255
01	Subcircuits	1550
02	Garage door	34
03	Kitchen aid chopper	16
04	Refrigerator	1212
05	A/V system of the living room	24
06	Computer 1	105
07	Laptop 1	16
08	Basement receiver/DVR/Blu-ray Player	79
09	Air compressor	42
10	LCD monitor 1	199
11	Television	151
12	Printer	364
13	Hair dryer	8
14	Iron	107
15	Empty socket of the living room	4
16	Monitor 2	481
TOTAL		**5647**

Additionally, although there are some devices that have better features to recognize daily routines that can be modified in order to improve consumption efficiency, such as the devices that are used frequently and operated manually, it has been considered that being able to identify all of them would allow to discard the less significant ones in a later process. The recognition of daily routines is one of the tasks proposed to continue the present study as discussed in Chapter 5.

Apart from that, for the validation of the model its quality must be measured. In the field of classification problems, and specifically those based on supervised learning algorithms, the confusion matrix is a representation of the model performance that can provide this information visually and effortlessly understood. The structure of the matrix has two dimensions: each row corresponds to a real class and each column corresponds to a predicted class. To fill out the resulting matrix, the correct and incorrect estimate counts are entered so that the prediction numbers are placed in the expected row and the column predicted for that class value. Listed below are the metrics of the confusion matrix.

- **True Positive (TP)**

The number of predictions where the classifier correctly predicts the corresponding class.

- **True Negative (TN)**

The number of estimates in which the classifier correctly predicts the rest of the classes.

- **False Positive (FP)**

The number of estimates in which the classifier incorrectly predicts the rest of the classes that are not considered as the class under study.

- **False Negative (FN)**

The number of predictions in which the classifier incorrectly predicts the corresponding class as another class that is not being considered in this iteration.

Figure 17 shows an example of an N-class classification problem. In this case, it is necessary to calculate the TP, TN, FP, and FN for each individual class. For instance, looking at a specific class Ct, the green element of the matrix represents the TP, whereas the orange elements correspond to the FP, the TN are coloured with magenta, and the FN with blue.

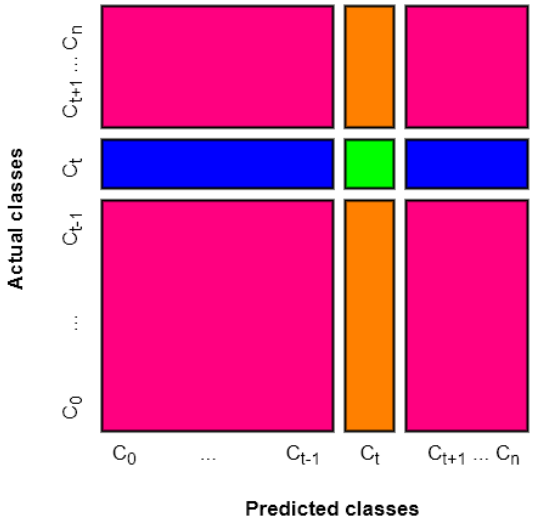


Figure 17. Confusion matrix for multi-class classification.

Although confusion matrix may describe the complete performance of the defined models, certain performance metrics of the models have been used to express it with numbers, which would be compared easily in the discussion section. Some of the most common performance metrics are:

- **Accuracy**

The accuracy relates the number of correctly classified to the overall number of predictions:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \in [0,1] \tag{4.1}$$

It is worth mentioning that it is a predominant metric, however, it may produce misleading results if in the dataset the number of samples from different classes varies greatly; in other words, when it is unbalanced. In those cases, the F_β-score works in a more appropriate way (G. M. Weiss, 2013).

- **Precision**

The precision expresses the classifier's certainty of correctly predicting a given class relating the amount of truly positive predicted samples to the total where this specific class was predicted:

$$Precision = \frac{TP}{TP + FP} \in [0,1] \tag{4.2}$$

- **Recall**

The recall, also called sensitivity, is similar to precision, but in this case, it compares the truly positive predicted samples with the total predictions where the class actually results:

$$Recall = \frac{TP}{TP + FN} \in [0,1] \tag{4.3}$$

- **F_β-score**

The F_β-score presents the classifier's ability to predict a given class taking into consideration both precision and recall:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \in [0,1] \tag{4.4}$$

The typical values of β are 1, 2 or 0,5. For the following tests a value of 1 is used for β.

On another note, this type of classification algorithms requires large amounts of data for a proper training, as will be discussed later. For this reason and taking into account that there may not be enough events for certain devices, and this can lead to an overfit, data augmentation techniques are used. In this case, introduction of noise has been used to include synthetic data and thus increase the size of the dataset. Furthermore, by doing so and using these new synthetic inputs together with the real ones obtained from the public database, the network could achieve a more generalized behaviour. The selected process is the introduction of noise due to the fact that the baseline data are time series, and this is the usual procedure for this type of data. For instance, there are other procedures such as introducing an offset value or displacing the point of interest through the temporal axis. These procedures had been discarded because, on the one hand, the offset value could conflict with the normalization process of the pre-processed stage or, if previously, would have no effect; and, on the other hand, the displacement would affect the position of the event within the temporal window but not the information that could be extracted from the signal.

As for the type of noise, the one selected was white Gaussian noise, which is a special case of Gaussian noise. In general, this type of noise is statistical, which means that its values follow a normal distribution, also called the Gaussian distribution. The specification, introduced by the colour white, refers to the power distribution, which in this case is uniform in a certain frequency band. For the process of the introduction of this noise the power of the signal is assumed to be 0 dBW. The signal-to-noise ratio (SNR), defined as the ratio between signal strength and noise power (A. B. Carlson, 2002), is set within the range between 1 dB and 40 dB. Likewise, to determine the appropriate noise level based on the value of SNR, the signal level is computed. In particular, the signals or samples to which this noise is introduced are some of the less numerous classes (all this to balance the distribution of samples). Table 5 presents the distribution of events per classes for each architecture after all the processing performed in the pre-processing stage and the creation of synthetic signals for data augmentation. Marked in orange are the values that differ from the number of events obtained from the database. As can be observed, some modifications have been introduced in the data augmentation process during the validation of the different architectures due to the final influence of this noise on the input samples of the ANNs. In addition, the distribution is not yet balanced, however, this mismatch could be solved with other variations that will be discussed later.

Table 5. Distribution of events per classes after data augmentation.

LABEL	NUMBER OF EVENTS		
	Recurrent based on LSTM	*Convolutional*	
		Time Sequence Images	*STFT Images*
00	**6324**	1255	1255
01	1550	1550	1550
02	**152**	**1740**	**716**
03	**60**	**800**	**336**
04	**6152**	1212	1212
05	**120**	**1248**	**504**
06	**576**	**5324**	105
07	**72**	**760**	**336**
08	**336**	**4116**	79
09	**220**	**2100**	**884**
10	**952**	**9992**	199
11	**728**	**7720**	151
12	**1908**	**18768**	364
13	**48**	**392**	**168**
14	**548**	**5400**	107
15	**24**	**204**	**0**
16	**2392**	**24600**	481
TOTAL	**22036**	**87148**	**8452**

The total set of events, whether captured in temporal window or in grayscale images, has been divided into three different subsets with its own use, in order to prepare an algorithm to make predictions.

- **Training data**

Training data is used for the learning process. This subset includes a vector or scalar that represents the corresponding output for each input. This value will be compared with the one obtained in order to learn. In this case, this value is the class tag of the event. The model evaluates the data repeatedly while adjusting its own parameters in different ways depending on the result of the comparison and the specific learning algorithm used. The term "parameters" refers to the coefficients of the model, that is to say, the weight and bias of the different activation functions and kernels.

- **Validation data**

During training, validation data is used to evaluate each adjusted model with new data, so it can be considered as a first test against unseen data to provide an evaluation of the model, while optimizing its hyperparameters. In this other case, the concept of hyperparameter corresponds to other elements of the model whose value is established before starting the learning process. For instance, some of the hyperparameters used in deep learning are: the number of hidden layers or neurons of the neural network; the Learning Rate, which refers to the step of backpropagation; the activation functions located at any point in the neural network; the batch size corresponds to the size of the sample which is training the model with, the number of epochs or times the algorithm will be training on the whole dataset; among others. Some of them had been mentioned before, and the others will be discussed later. Furthermore, it

should be noted that this data is not always used, although it may prevent the algorithm from memorizing inputs and outputs during training; in other words, it will help avoid overfitting. This last concept will be explained thereupon.

- **Test data**

Once the model is built, the test data validates that it can predict with enough performance, confirming that the neural network was trained effectively. Unlike validation data, this subset provides the final assessment. In addition, whereas training and validation data includes labels that will contribute to monitoring model's performance metrics, test data should not be labelled.

The distribution of the events (Figure 18) of the large initial dataset has been made so that 50 % of the available data will be used for the training process, half of the rest (25 %) corresponds to the validation data and the other half (25 %) will be used as test data. The division was carried out without repeating any events within the same subset or within others.

Figure 18. Distribution of the dataset.

It is important to note that the size of the dataset and its quality have some relevance during model training. The effects on model performance and its interaction with model capacity are summarized below:

- **Underfitting**

When the performance on the training data is poor since the capacity of the model is not enough to correctly perform its functionality and, in addition, it is not able to generalize to new data. The reason is that the degree of non-linearity in the data is higher than the amount of non-linearity the model is capable of capturing. An easy way to detect the underfitting (Figure 19) is by plotting the training and validation errors. Those values will closely follow each other and flatten around a large error value over a growing number of epochs. To solve this problem, the key idea is to increase the capacity of the model. To do this, there are several alternatives: increase the complexity of the model, change the regularization parameters or the Learning Rate or, finally, improve the quality of the data.

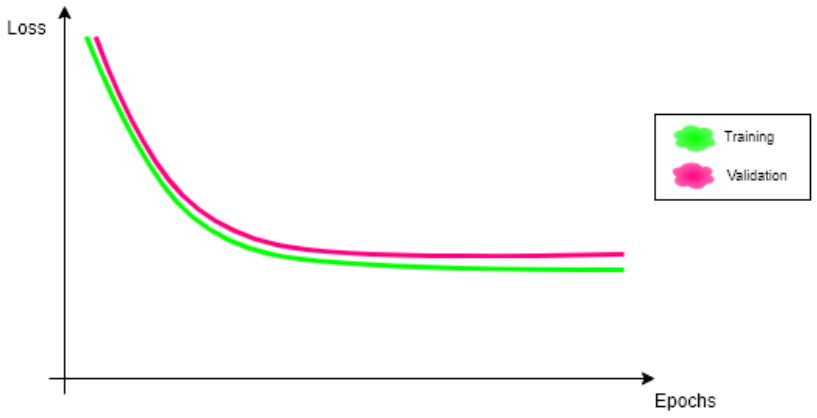


Figure 19. Example of how underfitting can be detected.

- **Overfitting**

When the model gets a low training data error, whereas the results in the validation data are poor. So, the performance in the training data is good, but the generalization to other data is poor. To illustrate the detection of overfitting, training and validation accuracies will be plotted (Figure 20). The accuracy of the training will increase with the number of epochs while the validation accuracy will diverge and stabilize at a much lower value, which will open a large gap between the two curves. In this case, more data should be included or, if this is not feasible, the model capacity should be reduced in order to adjust it to the data, or regularization should be used to compensate for the lack of data.

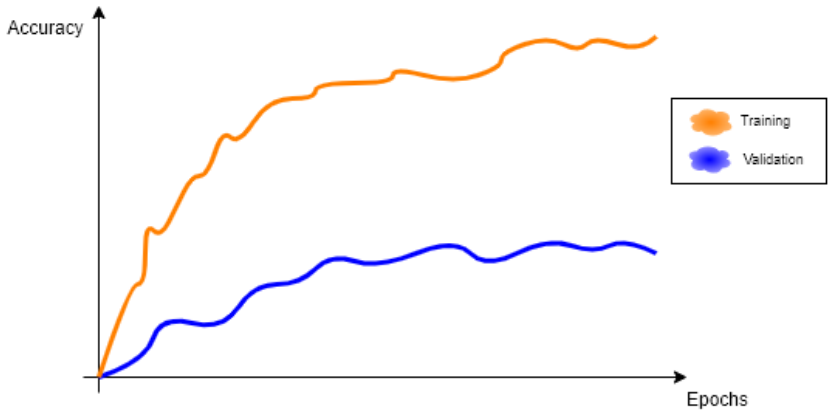


Figure 20. Example of how overfitting can be detected.

Lastly, it is turn for the training parameters settings. These factors will be used to control the training process and certain properties of the resulting model. A summary of the most relevant is listed below.

- **Initialization method**

The first step that must be considered is the initialization of parameters (M. V. Narkhede et al., 2022). Through this process, the shortest time will be used to optimize the performance of the structure. There are some techniques generally used to this issue, such as Zero initialization or Random initialization. However, those methods present certain problems associated with the training duration and the vanishing gradient problem. In order to solve these issues, other approaches had been proposed: for instance, He initialization or Xavier Initialization. As a default, Xavier or Glorot initialization is used for weights, whereas Zero initialization is used for bias.

  - ***Zero Initialization***

  This method sets all bias to zeros.

  - ***Xavier Initialization or Glorot Initialization***

  Xavier proposed a method (X. Glorot & Y. Bengio, 2010) that is calculated as a random number with a uniform probability distribution U between the specific range shown in (4.5), where n is the number of inputs to the node. In this method the weights such as the variance of the activations are the same across every layer. This will prevent the gradient from exploding or vanishing.

$$W = U\left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right] \tag{4.5}$$

- **Learning Rate**

The Learning Rate is acknowledged to be the most important hyperparameter due to the fact that, in general, an appropriate value for this parameter will maximize the model capacity. However, the problem is that if it is set to a large value, the training error can increase; while if set to a small value, it can cause the gradient descent to get stuck at a non-optimal point and, in addition, a slowdown in the training process. Thus, the best way to set a value for this parameter is to use a large value in the initial stages and reducing the Learning Rate as the system approaches the minimum. In this case, the initial Learning Rate is 0.0001, which will be varied during the process using a decay rate of 10^{-6}.

- **Learning algorithm**

Another strategy that is usually put into practice in this type of systems is to adapt the Learning Rate for each parameter thanks to the so-called learning algorithms (S. Ruder, 2016), such as Momentum, Nesterov Momentum, Adagrad, RMSProp or Adam. Specifically, Adagrad, RMSProp and Adam algorithms, as the training progresses they automatically adapt the effective Learning Rate; whereas Momentum, Nesterov Momentum and Adam algorithms are able to improve the convergence speed. Finally, since Adam algorithm combines both features, it is the one used in these approaches.

  - ***Adaptive Moment Estimation (Adam)***

  The Adam algorithm equations (D. P. Kingma & J. Ba, 2015) are explained hereafter.

  1) Firstly, the sequence $\Lambda(n)$ is used to provide "Momentum" to the updates (4.6). This equation allows to obtain the values of the first moment estimation. In this equation, n is the number of iterations, $g(n)$ is the gradient evaluated and the exponential decay rate $\beta_1 \in [0,1)$ is usually defaulted to 0.9.

$$\Lambda(n) = \beta_1 \cdot \Lambda(n-1) + (1-\beta_1) \cdot g(n) \tag{4.6}$$

  2) Secondly, the sequence $\Delta(n)$ customizes the effective Learning Rate with respect to each parameter so that the rates for parameters with larger gradients equal those for parameters with smaller gradients (4.7). With this equation the values of the second moment are obtained. In this case, the exponential decay rate $\beta_2 \in [0,1)$ is usually defaulted to 0.999.

$$\Delta(n) = \beta_2 \cdot \Delta(n-1) + (1-\beta_2) \cdot g(n)^2 \tag{4.7}$$

  3) Additionally, to counteract that both sequences are initialized as vectors of 0, bias-corrected estimates are calculated (4.8 – 4.9).

$$\hat{\Lambda}(n) = \frac{\Lambda(n)}{1-\beta_1^n} \tag{4.8}$$

$$\hat{\Delta}(n) = \frac{\Delta(n)}{1-\beta_2{}^n} \tag{4.9}$$

  4) Finally, the parameter update equation is presented (4.10) where the step-size η is set to 0.1 and ε, which is used to avoid the division by zero, is usually defaulted to 10^{-8}.

$$\theta(n+1) = \theta(n) - \eta \cdot \frac{\hat{\Lambda}(n)}{\sqrt{\hat{\Delta}(n) + \varepsilon}} \tag{4.10}$$

In the approaches presented in this Master's Thesis, the values by default for β_1, β_2, η and ε are used.

- **Regularization techniques**

These methods are used to avoid overfitting. The way these algorithms work is to reduce the effectiveness of the models because in these situations it exceeds the requirements of the problem. The most common techniques (J. Kukačka et al., 2017) are Early Stopping, L2 Regularization, L1 Regularization, Dropout Regularization, Data Augmentation, and Batch Normalization. Among all of them, some data augmentation procedures have been used, as discussed above, but also early stopping techniques will also be used.

  - ***Early Stopping***

    The idea is to stop the training process when a monitored metric has stopped improving (Figure 21). At that point it is assumed that the model begins to overfit to the training data. The quantity to be monitored is validation loss, which is computed at the end of each training epoch. In practise, it is common to wait until loss has stopped decreasing for a number of epochs before stopping and establish a minimum change in the monitored quantity to qualify as an improvement. In this case, the selected number of epochs is five, but no minimum change is determined.

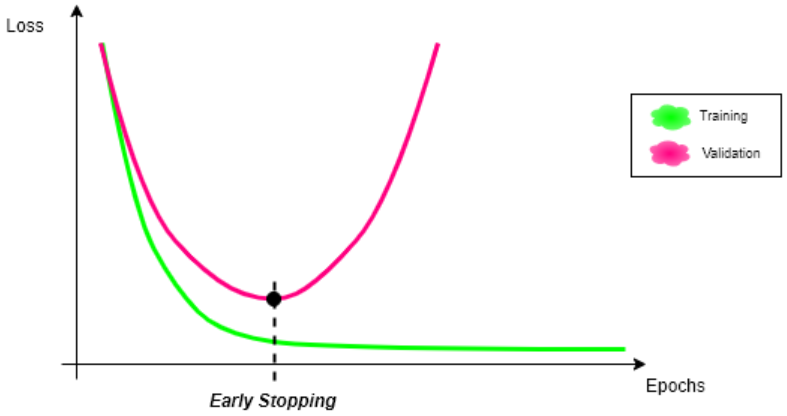


Figure 21. Example of how the early stopping can be represented.

- **Number of epochs**

During the training process each complete presentation of all the samples belonging to the training set is called an epoch. This concept specifies how many times it is necessary to present all the samples of the training set to adjust the parameters of the neural network. As stated above, one of the most critical issues is overfitting and, to solve it, early stopping is implemented, so the "number of epochs" parameter refers to the maximum. This ensures that, at least, the formation process will not be infinite if convergence does not occur. For each architecture this value will be different since they work with different samples that make the process of extracting information differ from each other (Table 6).

Table 6. Maximum number of epochs for each architecture.

ARCHITECTURE		EPOCHS
Recurrent based on LSTM		200
Convolutional	***Time Sequence Images***	200
	STFT Images	600

- **Batch size**

Training process will be carried out through iterations where a subset of the training set called batch or minibatch will be used. Minibatch size is a key factor that influences the system performance. For instance, large lots can provide a more accurate estimate, whereas small batches can offer a regularization effect. Nevertheless, large batches are limited by the hardware used to process the information, or at least this hardware achieves a better runtime with specific array sizes, whereas some deep architectures are underutilized by extremely small batches. Due to all these factors, the number of samples per batch was obtained experimentally and the optimize value was defined to one-hundredth of the amount of the training data.

- **Shuffling**

Another crucial aspect is that the minibatches must be selected at random so that not only the samples are independent within this set, but also subsequent minibatches are independent from each other. In practice, it is enough to shuffle the samples. Overall, this process does not appear to have a significant detrimental effect; however, not shuffling in any way can seriously reduce the effectiveness of the algorithm. In these approaches, training data is shuffled before each epoch.

4.2 Results of the recurrent architecture

For the recurrent architecture different tests have been performed in order to extract the possible limitations of the model:

- **Adding class weights**

First of all, as discussed above, since there is an imbalance in the number of samples that can affect the performance of the algorithm, different weights will be provided during training for the different classes. These weights work, so the highest value will be for the minority class and for the majority, the opposite.

- **Modifying the size of the structure**

Secondly, some structure size modifications have been made by varying the number of neurons in the layers of the ANN in order to see if it is possible to improve its performance. Specifically, increasing the size of the first LSTM layer by multiplying by 20 the number of units or reducing this size by dividing by 20.

- **Discard Class 01**

The last test involves discarding Class 01 samples. The reason is that within this class, which contains information about the distribution circuits, there could be events produced by other devices already present in the classification or even unknows.

Table 7 compares the results of the performance metrics obtained from the test stage, after training and validation. As this table shows, the different tests present similar results, so it can be concluded that the modifications do not represent a significant improvement. Specifically, for example, discarding circuits seems to improve metrics except accuracy, whereas the opposite happens when class weights are added. In addition, as for the modifications of the structure, a decrease in size implies a decrease in all metrics, whereas almost all improve by increasing the size.

Table 7. Experimental results of the recurrent architecture comparing different test.

CONFIGURATION	ACCURACY	PRECISION	RECALL	F1-SCORE
Main structure	99.75%	95.24%	97.41%	96.11%
Adding class weights	99.76%	94.97%	97.54%	95.97%
Increasing structure size	99.77%	96.79%	97.17%	96.93%
Reducing structure size	99.68%	90.66%	96.12%	92.50%
Discard Class 01	99.67%	96.29%	97.67%	96.93%

Furthermore, in order to visualize the results, Figure 22 represents the confusion matrix of the main structure. In this confusion matrix it can be perceived that most errors are concentrated in the rows and columns that refer to classes 00 and 01, which involve sets of several loads. Moreover, in the case of class 00, these events can occur at the same time as other devices of higher consumption and, therefore, the predominant event that will be recognized is that of the other device. Other mistakes, but less numerous, appear for classes 06, 07, 08 and 10.

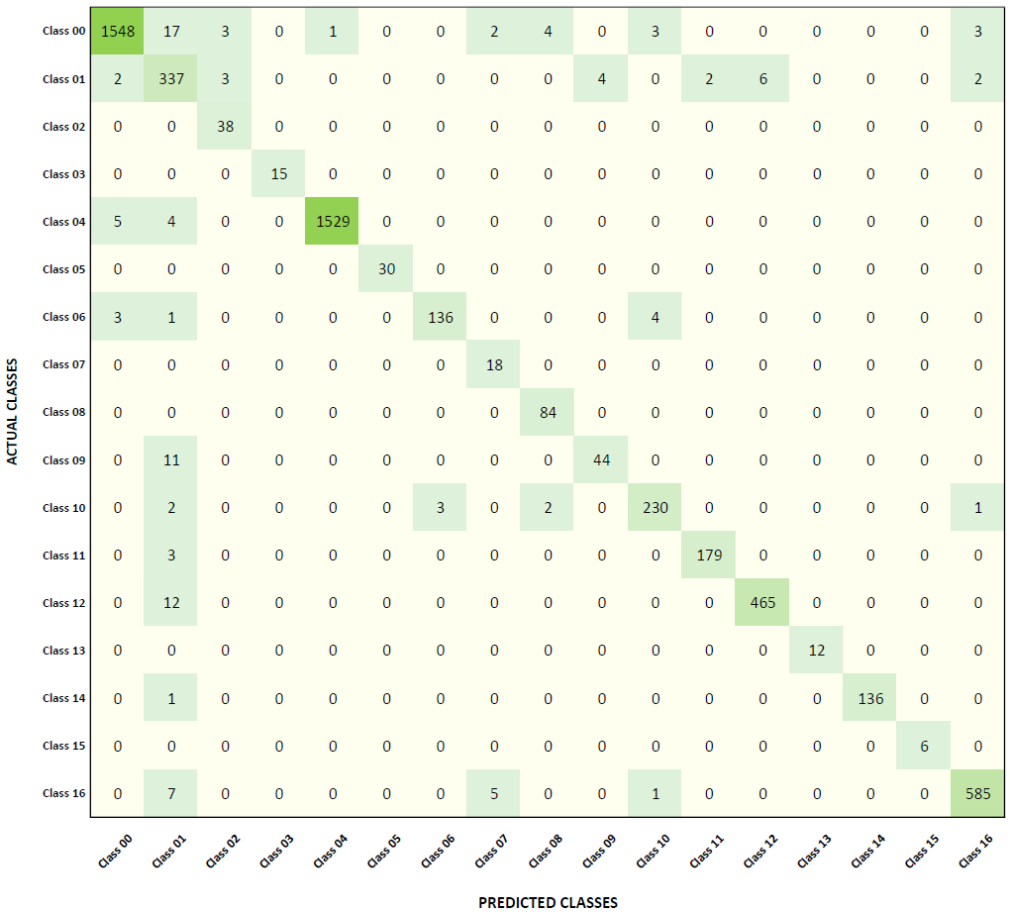


Figure 22. Confusion matrix of the recurrent architecture using the main structure of the network.

4.3 Results of the convolutional architectures

For the first convolutional architecture, instead of performing different tests, other versions of the network were previously defined, as shown in Figure 23. As reported below, the architecture defined in Chapter 3 was obtained as a result of the modifications made to these preceding versions.

- **Version 1**

The differences are that the number of filters used in convolutional layers is four times greater; that is, instead of 16 there are 64 and in the case of 32 they are 128; in addition, the dimensions of the cores are not odd; and finally, the Max Pooling layer is not included between the two convolutional layers.

The change to odd dimension is due to the fact that the use of odd size filters symmetrically divides the previous layer around the output pixel, avoiding possible distortions between the layers if this symmetry did not exist. Likewise, the goal of the interspersed Max Pooling layer is that, through the reduction of the size of the images filtered by the first convolution, the following layers reduce their size.

- **Version 2**

For this second version, the number of filters is reduced, and odd-sized cores are used.

- **Version 3**

This last version corresponds to the ANN described in Chapter 3. For this version the Max Pooling layer is introduced.

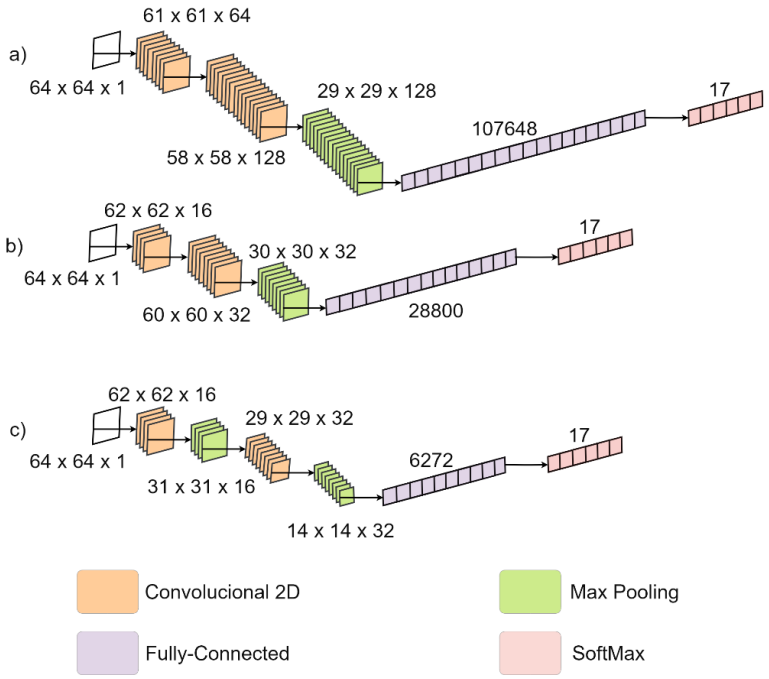


Figure 23. Structure of the proposed CNNs: (a) Version 1, (b) Version 2, and (c) Version 3 (L. de Diego-Otón et al., 2021).

In this case, the performance metrics resulting from these three versions are compared in Table 8. According to these results, Version 2 is the one that has obtained the best result, followed by Version 3 and, at the end, Version 1. Taking into account that the structure of Version 3 presents a reduction in the size of its layers with respect to Version 2, the conclusion reached is that the network offers results of great accuracy and precision, being smaller.

Table 8. Experimental results of the first convolutional architecture comparing different versions.

CONFIGURATION	ACCURACY	PRECISION	RECALL	F1-SCORE
Version 1	99,01 %	68,75 %	77,44 %	70,78 %
Version 2	99,71 %	92,99 %	92,82 %	92,85 %
Version 3	99,70 %	92,17 %	91,78 %	91,91 %

With respect to the confusion matrix, shown in Figure 24, the same as in the previous case, most of the errors are found for classes 00 and 01. Moreover, although some other events are misclassified, the largest FN and FP values are for those classes.

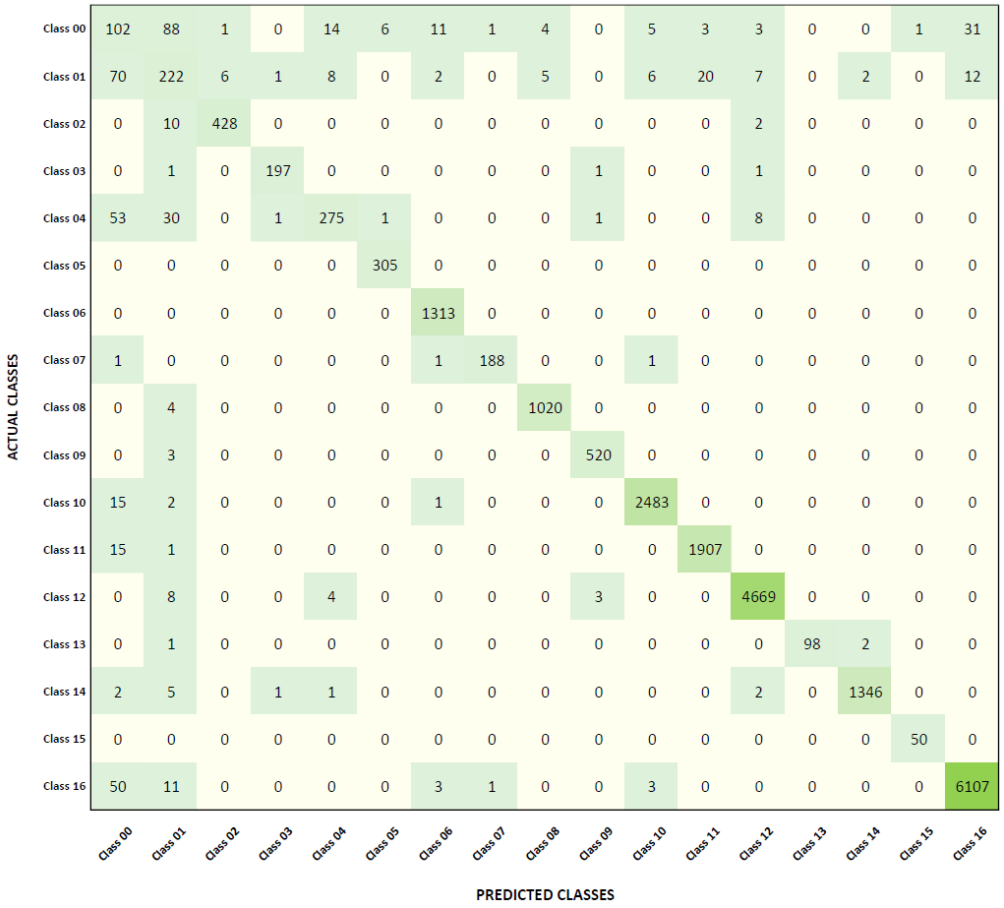


Figure 24. Confusion matrix of the definitive version of the first convolutional architecture.

For the convolutional architecture which works with the frequency features, the distinction of the two ways of representing the information obtained from the STFT of the images could be contrasted and in Table 9 these metrics are represented. Between both configurations, the alternative of using magnitude and phase presents better results, in general, than the real and imaginary part option. However, with respect to the results discussed above, this second convolutional architecture reaches the worst values for the performance metrics. Moreover, it is worth to highlighting that, with regard to the electrical devices of the classification, the empty socket of the living room (class 15) has been excluded because the information provided by its events was considered irrelevant.

Table 9. Experimental results of the second convolutional architecture comparing the information of different input samples.

CONFIGURATION	ACCURACY	PRECISION	RECALL	F1-SCORE
Magnitude and Phase	97,17 %	79,93 %	66,39 %	70,34 %
Real and Imaginary part	96,91 %	73,03 %	67,07 %	68,45 %

In this case, both confusion matrices, Figure 25 and Figure 26, are included in the manuscript. In them it can be observed that not only the classes 00 and 01 have great number of errors, but also other classes such as 02, 03, 05, 07, 10, 13 or 16 have some misclassified events.

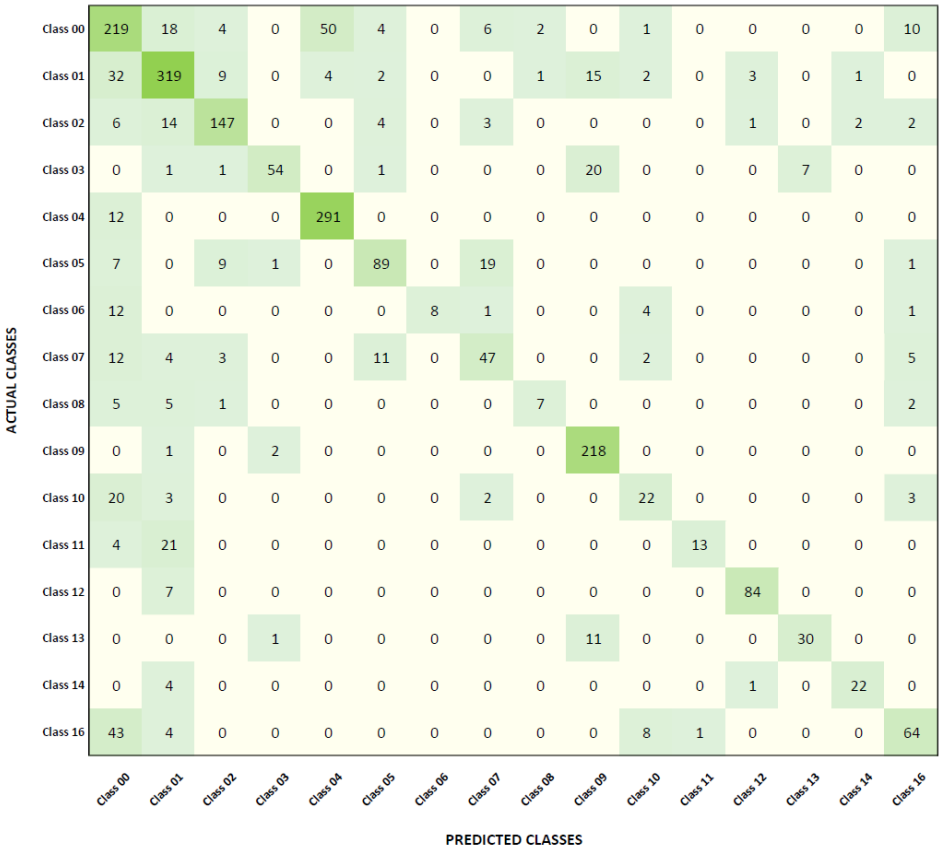


Figure 25. Confusion matrix of the second convolutional architecture distinguishing between magnitude and phase.

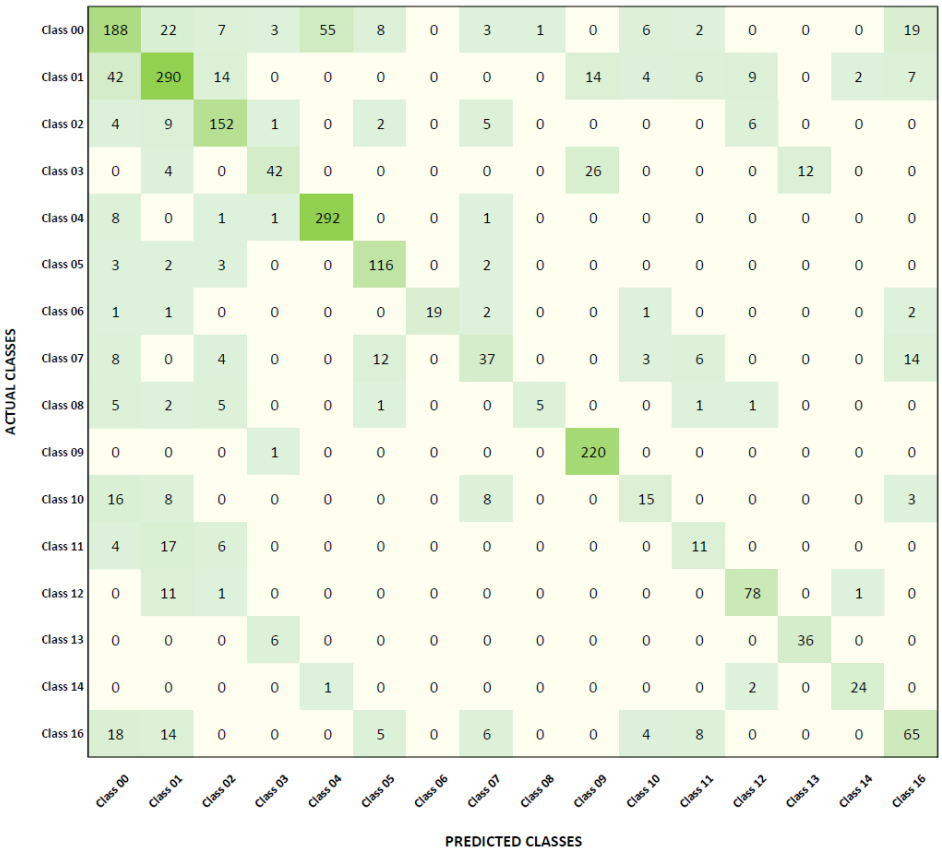
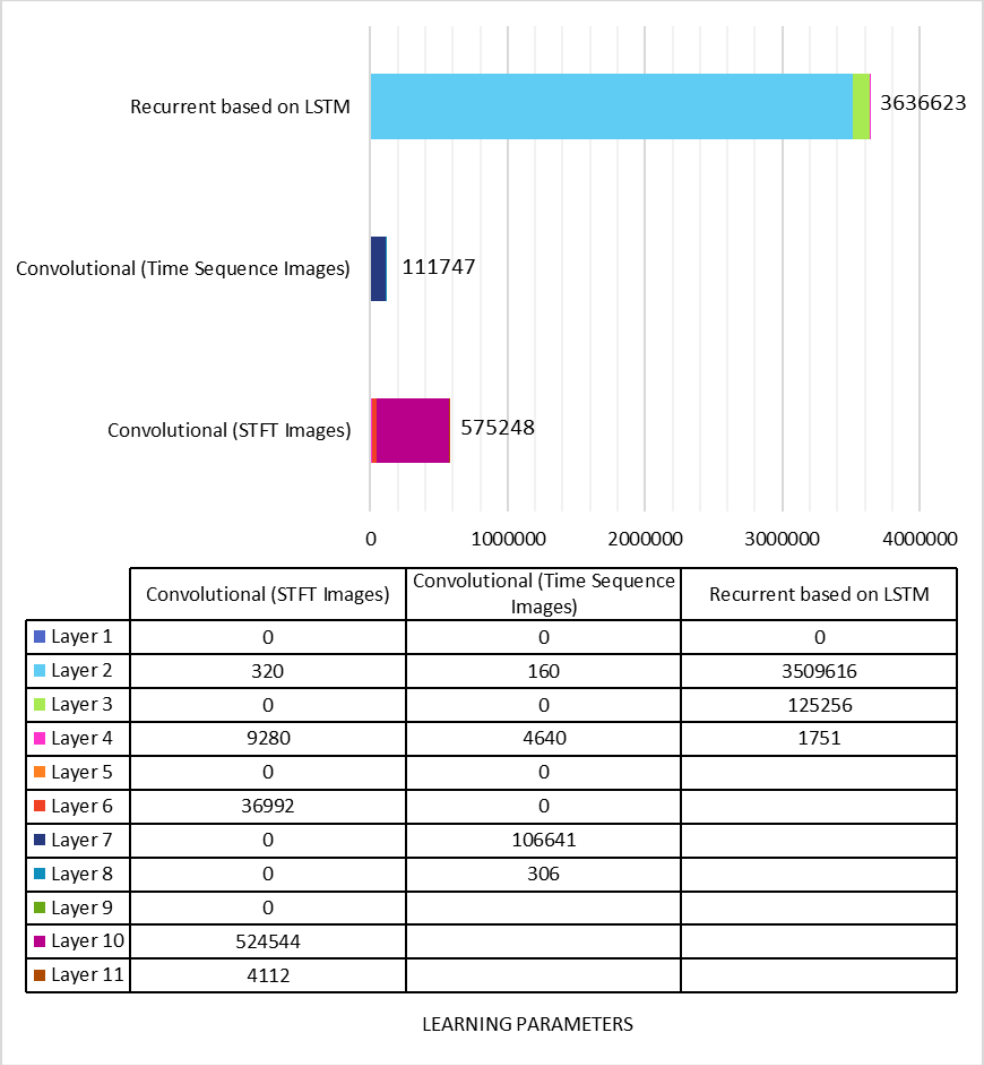


Figure 26. Confusion matrix of the second convolutional architecture distinguishing between real and imaginary part.

4.4 Discussion

After having detailed the classification performance of each of the proposed architectures, it is time to make a comparison between them. To do so, not only the values of the performance metrics are important, but it is also worth considering the computational complexity, as has already been commented slightly in some comparison in the previous section. Analyzing network complexity is a key part of the design process because this dimensionality can affect learning ability. However, it is even more important for implementation because, before it can be used in any application, it is necessary to take this into account to assess the required storage. This concept will be represented by the internal configuration of the model that can be measured through the number of learning parameters which in turn depends on the number and dimensions of the layers. The contribution of each layer and the total number of learning parameters for each approach is provided in Figure 27. As can be observed, the recurrent architecture has the highest number of learning parameters, whereas the parameters of convolutional structures are even less than a quarter of that number. Between convolutional structures, the size of the input images and the structure itself causes the difference of the values.



	Convolutional (STFT Images)	Convolutional (Time Sequence Images)	Recurrent based on LSTM
Layer 1	0	0	0
Layer 2	320	160	3509616
Layer 3	0	0	125256
Layer 4	9280	4640	1751
Layer 5	0	0	
Layer 6	36992	0	
Layer 7	0	106641	
Layer 8	0	306	
Layer 9	0		
Layer 10	524544		
Layer 11	4112		

LEARNING PARAMETERS

Figure 27. Number of learning parameters for all three architectures[1].

On the other hand, for a further and more detailed evaluation of the classification performance, Figure 28 represents the experimental metrics of all the proposed architectures. It should be underlined that this comparison takes into account the two ways of representing the images resulting from the STFT process. This graph shows that, whereas the recurrent method obtains better results in all metrics, followed by the convolutional that uses time sequence images, the architecture that uses the images resulting from the STFT process achieves the lowest values. As a conclusion, all these arguments show that the architecture that used the convolutional network to classify the time sequence images has more interesting features: the high classification performance along with the lowest complexity.

[1] Keep in mind that the recurrent structure has four layers, convolutional structure which works with temporal sequence images has eight layers and, finally, the other convolutional one has eleven layers.

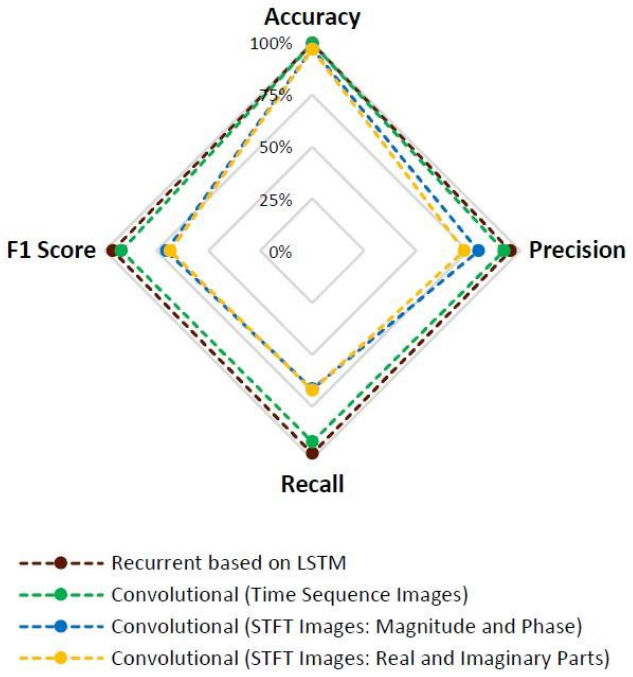


Figure 28. Comparison of the performance metrics of all the architectures (L. de Diego-Otón et al., 2022).

Finally, some considerations concerning the electrical devices and samples used in the classification will be discussed. Firstly, with regard to the devices of the classification, the distribution circuits, which have been discarded on occasion due to their strong negative influence on the results, will be excluded together with the empty plugs. Only events from well-defined electrical devices will be used. Furthermore, in previous works, such as (F. Ciancetta et al., 2021) or (S. Houidi et al., 2020), the classification is carried out by considering that there are different state transitions for each device, such as switching on and off. This distinction can benefit the results because they would be treated as distinct classes. However, the problem is when the device has multiple modes of operation since there should be a class for each transition change between them.

Furthermore, regarding to the data augmentation, the definition of a specific process which allows to obtain synthetic samples of great value will be sought. In parallel, other databases where there are more houses should be used since the BLUED dataset only contains one household. This should help the generalization of neural networks.

Chapter 5 Conclusions and future works

Through this chapter, the objectives will be reviewed to explore the extent to which they have been achieved and how this research has contributed to the current state-of-art. After this, some of the open issues out of the scope of this thesis, which deserve further research, will be presented.

5.1 Conclusions

In this Master's Thesis, it is addressed the problem of the classification of domestic loads on the basis of information about the aggregate consumption of a residence. One of the main contributions of this work is to propose methods to solve it based on ANNs. Specifically, a comparative evaluation of the disaggregation capacity of three different defined architectures has been provided, considering as a signal of interest the electric current sampled at high frequency.

In order to do so, it was essential to carry out a previous study of the topic related backgrounds. The objective of this analysis was to define the bases that would allow to choose, between the multiple algorithms, those alternatives that, although capable of offering high performance, present the least possible computational complexity to be implemented in low-cost hardware platforms. Apart from the theoretical basis, a fundamental element was the conformation of the proper experimental framework. This ranges from the selection of the database that contains the necessary information to carry out the validation of the algorithms to the configuration of the training parameters that will influence certain properties of the resulting model.

The proposed architectures were defined as event-based supervised solutions where the electrical signal is not only processed, but also the input samples of the ANNs are captured around the position of the events located in this treated signal. These samples conform temporal windows that could be used directly, as in the first approach, or they must be transformed into images, as in the cases of the second and third algorithms. For the second one, images are constructed by creating the rows of a matrix from divisions of that temporal window; and for the third one, they are obtained by using the STFT process. In other words, for the creation of the images, both the time and frequency domain have been considered. Furthermore, the intelligent algorithms included in those architectures are: for the first approach, an RNN based on LSTM cells; and for the second approach, a CNN.

From an experimental point of view, the validation was performed using the data provided by the public database called BLUED. The measurement variables have been collected at a frequency of 12 kHz for a week. Results show that the proposals which use the recurrent topology and the convolutional one, whose inputs are time sequence images, obtain better results than the convolutional architecture that classifies based on the characteristics of the frequency domain. These differences in the results have been also compared considering the computational complexity of the alternatives which will give an idea of the feasibility of implementation on real-time platforms. In this context, the second architecture, namely the one that used

the convolutional network to classify the time sequence images, achieves adequate classification performance whereas its complexity is the lowest.

On the other hand, in collaboration with other members of the GEINTRA Research Group, a smart meter prototype has been developed (R. Nieto et al., 2021). The proposal consists of a System-on-Chip (SoC) architecture based on a Xilinx-7000 Zynq FPGA (*"Field-Programmable Gate Array"*). It has been tested in the laboratory, where different appliances such as a fan heater, a hair dryer, a coffee machine, and a microwave are monitored. The acquisition process is carried out at 4 ksps using the ADE9153A integrated circuit (Analog Devices, 2018). However, for the identification of the load and the final dissemination of the results, it is used a remote server that has been implemented in a Raspberry Pi 2, which integrates a Quad-Core ARM Cortex-A7 (R. Reed et al., 2015). The communication in this smart meter is provided by the Wi-Fi adapter (Digilent, 2016). Additionally, it is worth mentioning that only the data related to each event is sent upstream to this cloud server, so a reduction in the bandwidth required in cloud communication is made. Lastly, the load identification results are published on the Thingspeak™ platform.

5.2 Future Works

Future works concern deeper analysis in some specific developments to improve and support the proposed architecture, likewise, it is also included the real time implementation and the routines extraction algorithm definition. These last points, since they exceed the scope of this Master's Thesis, have not been discussed in the manuscript, but they are part of the initial focus of the research project.

- **Unsupervised architectures**

This Master's Thesis has been mainly focused on supervised learning architecture, but it could be interesting to become closer to a real situation, where the useful life of appliances is limited, and they are often replaced. For that reason, unsupervised learning architectures seem to be more useful because they are able to learn without prior information, so the replacement of appliances would not involve a retraining process.

- **Higher sampling frequencies**

Increasing the sampling rate would allow the system to be able to detect switched-mode power supplies which operate at frequencies ranging from several hundred kHz to several MHz. Specifically, these power supplies are widely used in a variety of electronic equipment, including computers and other sensitive systems, which require a stable and efficient power supply.

- **Real-time implementation**

The implementation of the current proposal in real time and with low-cost hardware platforms is a point in which work is being done and will be worked on within the research project.

- **Behavioural routines**

Last but not least, one of the lines for which the research project, discussed at the beginning of this manuscript, will continue is the creation of algorithms capable of extracting behavioural routines from the detection of the use of different electrical devices. With information on the consumption behaviour of consumers, personal optimization of their energy consumption could be offered in order to allow substantial savings, as well as the other applications commented in Chapter 2.

Bibliography

A. B. Carlson. (2002). Communication system: An introduction to signals and noise in electrical communication (4th ed.). McGraw-Hill series in electrical engineering.

A. Dehghani, O. Sarbishei, T. Glatard, & E. Shihab. (2019). A Quantitative Comparison of Overlapping and Non-Overlapping Sliding Windows for Human Activity Recognition Using Inertial Sensors. Sensors (Basel, Switzerland), vol. 19(22), p. 5026. 10.3390/s19225026.

A. F. Agarap. (2018). Deep Learning using Rectified Linear Units (ReLU). arXiv, abs/1803.08375, n.p. 10.48550/arXiv.1803.08375.

A. Graves, A. Mohamed, & G. Hinton. (2013). Speech recognition with deep recurrent neural networks. In the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645-6649. 10.1109/ICASSP.2013.6638947.

A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro, & A. M. Tonello. (2014). GREEND: An energy consumption dataset of households in Italy and Austria. In the IEEE International Conference on Smart Grid Communications (SmartGridComm'14), pp. 511-516. 10.1109/SmartGridComm.2014.7007698.

A. Moradzadeh, O. Sadeghian, K. Pourhossein, B. Mohammadi-Ivatloo, & A. Anvari-Moghaddam. (2020). Improving Residential Load Disaggregation for Sustainable Development of Energy via Principal Component Analysis. Sustainability, vol. 12(8), sp. 3158. 10.3390/su12083158.

A. Patle, & D. S. Chouhan. (2013). SVM kernel functions for classification. In the International Conference on Advances in Technology and Engineering (ICATE'13), pp. 1-9. 10.1109/ICAdTE.2013.6524743.

A. R. Rababaah, & E. Tebekaemi. (2012). Electric load monitoring of residential buildings using goodness of fit and multi-layer perceptron neural networks. In the IEEE International Conference on Computer Science and Automation Engineering (CSAE'12), , vol. 2 pp. 733-737. 10.1109/CSAE.2012.6272871.

A. Reinhardt, P. Baumann, D. Burgstahler, M. Hollick, H. Chonov, M. Werner, & R. Steinmetz. (2012). On the accuracy of appliance identification based on distributed load metering data. In the Sustainable Internet and ICT for Sustainability (SustainIT'12), pp. 1-9.

A. Ruano, A. Hernandez, J. Ureña, M. Ruano, & J. Garcia. (2019). NILM Techniques for Intelligent Home Energy Management and Ambient Assisted Living: A Review. Energies, vol. 12(11), sp. 2203. 10.3390/en12112203.

A. S. Hernández, A. H. Ballado, & A. P. D. Heredia. (2021). Development of a Non-Intrusive Load Monitoring (NILM) with Unknown Loads using Support Vector Machine. In the IEEE International Conference on Automatic Control & Intelligent Systems (I2CACIS'21), pp. 203-207. 10.1109/I2CACIS52118.2021.9495876.

A. S. Siddiqui, & A. M. Sibal. (2020). Energy Disaggregation in Smart Home Appliances: A Deep Learning Approach. Energy, n.p. Available online in https://hal.archives-ouvertes.fr/hal-02954362

A. Yasin, & S. A. Khan. (2018). Unsupervised Event Detection and On-Off Pairing Approach Applied to NILM. In the International Conference on Frontiers of Information Technology (FIT'18), pp. 123-128. 10.1109/FIT.2018.00029.

Analog Devices. (2018). Data Sheet ADE9153A. Energy Metering IC with Autocalibration. Available online in: https://www.analog.com/media/en/technical-documentation/data-sheets/ade9153a.pdf

B. Bertalanic, G. Cerar, M. Meza, & C. Fortuna. (2021). Designing a Machine Learning based Non-intrusive Load Monitoring Classifier.

B. Larson, L. Gilman, R. Davis, M. Logsdon, J. Uslan, B. Hannas, D. Baylon, P. Storm, V. Mugford & N. Kvaltine. (2014). Residential building stock assessment: Metering study. Northwest Energy Effic.Alliance (NEEA). Available online in: https://neea.org/data/residential-building-stock-assessment

B. Mahapatra, & A. Nayyar. (2019). Home energy management system (HEMS): concept, architecture, infrastructure, challenges and energy management schemes. Energy Systems, n.p. 10.1007/s12667-019-00364-w.

B. Najafi, S. Moaveninejad, & F. Rinaldi. (2018). Chapter 17 - Data Analytics for Energy Disaggregation: Methods and Applications. Big Data Application in Power Systems (pp. 377-408). Elsevier. 10.1016/B978-0-12-811968-6.00017-6.

C. Beckel, W. Kleiminger, R. Cicchetti, T. Staake, & S. Santini. (2014). The ECO data set and the performance of non-intrusive load monitoring algorithms. In the Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys '14), pp. 80-89. 10.1145/2674061.2674064.

C. C. Yang, C. S. Soh, & V. V. Yap. (2017). A non-intrusive appliance load monitoring for efficient energy consumption based on Naive Bayes classifier. Sustainable Computing: Informatics and Systems, vol. 14, pp. 34-42. 10.1016/j.suscom.2017.03.001.

C. C. Yang, C. S. Soh, & V. V. Yap. (2018). A systematic approach in appliance disaggregation using k-nearest neighbours and naive Bayes classifiers for energy efficiency. Energy Efficiency, vol. 11(1), pp. 239-259. 10.1007/s12053-017-9561-0.

C. Gisler, A. Ridi, D. Zufferey, O. A. Khaled, & J. Hennebert. (2013). Appliance consumption signature database and recognition test protocols. In the 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA'13), pp. 336-341. 10.1109/WoSSPA.2013.6602387.

C. Klemenjak, C. Kovatsch, M. Herold, & W. Elmenreich. (2020). A synthetic energy dataset for non-intrusive load monitoring in households. Scientific Data, vol. 7, sp. 108. 10.1038/s41597-020-0434-6.

D. Christensen, L. Earle, & B. Sparn. (2012). NILM Applications for the Energy-Efficient Home. Contract (New York, N.Y. : 1960), vol. 303, pp. 275-3000.

D. G. Kleinbaum, & M. Klein. (2002). Logistic Regression: A Self-Learning Text (3th ed.). Blackwell Publishing.

D. Murray, L. Stankovic, & V. Stankovic. (2017). An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study. Scientific Data, vol. 4(1), sp. 160122. 10.1038/sdata.2016.122.

D. P. Kingma, & J. Ba. (2015). Adam: A method for stochastic optimization. CoRR, abs/1412.6980, n.p.

D. P. Kingma, & M. Welling. (2019). An Introduction to Variational Autoencoders. 10.1561/9781680836233.

D. P. Renaux, F. Pottker, H. C. Ancelmo, A. E. Lazzaretti, C. R. Lima, R. R. Linhares, E. Oroski, L. D. Nolasco, L. T. Lima, B. M. Mulinari, J. R. Silva, J. S. Omori, & R. B. Santos. (2020). A Dataset for Non-Intrusive Load Monitoring: Design and Implementation. Energies, vol. 13(20), sp. 5371. 10.3390/en13205371.

Digilent, I. (2016). Pmod WiFi Reference Manual. Available online in: https://reference.digilentinc.com/_media/reference/pmod/pmodwifi/pmodwifi_rm.pdf

F. Ciancetta, G. Bucci, E. Fiorucci, S. Mari, & A. Fioravanti. (2021). A New Convolutional Neural Network-Based System for NILM Applications. In the IEEE Transactions on Instrumentation and Measurement, , vol. 70 pp. 1-12. 10.1109/TIM.2020.3035193.

F. Rosenblatt. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, vol. 65(6), pp. 386-408. 10.1037/h0042519.

G. Bucci, F. Ciancetta, E. Fiorucci, S. Mari, & A. Fioravanti. (2021). State of art overview of Non-Intrusive Load Monitoring applications in smart grids. Measurement: Sensors, vol. 18, sp. 100145. 10.1016/j.measen.2021.100145.

G. D. Forney. (1973). The viterbi algorithm. 10.1109/PROC.1973.9030.

G. F. Angelis, C. Timplalexis, S. Krinidis, D. Ioannidis, & D. Tzovaras. (2022). NILM applications: Literature review of learning approaches, recent developments and challenges. Energy and Buildings, vol. 261, sp. 111951. 10.1016/j.enbuild.2022.111951.

G. M. Weiss. (2013). Foundations of Imbalanced Learning. In H. Haibo, & Y. Ma (Eds.), (pp. 13-41) 10.1002/9781118646106.ch2.

G. P. H. Styan. (1973). Hadamard products and multivariate statistical analysis. Linear Algebra and its Applications, vol. 6, pp. 217-240. 10.1016/0024-3795(73)90023-2.

H. Grover, L. Panwar, A. Verma, B. K. Panigrahi, & T. S. Bhatti. (2022). A Multi-Head Convolutional Neural Network Based Non-Intrusive Load Monitoring Algorithm Under Dynamic Grid Voltage Conditions. arXiv, abs/2205.15994, n.p. 10.48550/arXiv.2205.15994.

H. Kim. (2011). Unsupervised Disaggregation of Low Frequency Power Measurements. In the Proceedings of the SIAM International Conference on Data Mining (SDM'11), 11 pp. 747-758. 10.1137/1.9781611972818.64.

H. Rafiq, X. Shi, H. Zhang, H. Li, & M. K. Ochani. (2020). A Deep Recurrent Neural Network for Non-Intrusive Load Monitoring Based on Multi-Feature Input Space and Post-Processing. Energies, vol. 13(9), sp. 2195. 10.3390/en13092195.

I. Goodfellow, Y. Bengio, & A. Courville. (2016). Deep Learning. MIT Press.

J. Gao, S. Giri, E. C. Kara, & M. Bergés. (2014). PLAID: a public dataset of high-resoultion electrical appliance measurements for load identification research: demo abstract. In the Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys'14), pp. 198-199. 10.1145/2674061.2675032.

J. Kelly, & W. Knottenbelt. (2015). The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. Scientific Data, vol. 2(1), sp. 150007. 10.1038/sdata.2015.7.

J. Kolter, & M. Johnson. (2011). REDD: A Public Data Set for Energy Disaggregation Research. Artificial Intelligence, vol. 25, pp. 59-62. Available online in: http://redd.csail.mit.edu/

J. Kukačka, V. Golkov, & D. Cremers. (2017). Regularization for deep learning: A taxonomy. arXiv, abs/1710.10686. 10.48550/arXiv.1710.10686.

J. Lederer. (2021). Activation Functions in Artificial Neural Networks: A Systematic Overview. ArXiv, abs/2101.09957. 10.48550/arXiv.2101.09957.

J. P. Zimmermann, M. Evans, J. Griggs, N. King, L. Harding, P. Roberts, & C. Evans. (2012). Household Electricity Survey: A Study of Domestic Electrical Product Usage. Available online in: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/208097/10043_R66141HouseholdElectricitySurveyFinalReportissue4.pdf

J. Revuelta Herrero, Á. Lozano Murciego, A. Barriuso, D. Hernández de la Iglesia, G. Villarrubia González, J. M. Corchado Rodríguez, & R. Carreira. (2018). Non Intrusive Load Monitoring (NILM): A State of the Art. In the International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'18), pp. 125-138. 10.1007/978-3-319-61578-3_12.

J. Schultz, R. R. Copley, T. Doerks, C. P. Ponting, & P. Bork. (2000). SMART: a web-based tool for the study of genetically mobile domains. Nucleic Acids Research, vol. 28(1), pp. 231-234. 10.1093/nar/28.1.231.

J. Zheng, D. W. Gao, & L. Lin. (2013). Smart Meters in Smart Grid: An Overview. In the IEEE Green Technologies Conference (GreenTech'13), pp. 57-64. 10.1109/GreenTech.2013.17.

K. Bache, & M. Lichman. (2013). Individual Household Electric Power Consumption Dataset. Available online in: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

K. Cho, B. Van Merriënboer, D. Bahdanau, & Y. Bengio. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv, abs/1409.1259. 10.48550/arXiv.1409.1259.

K. D. Anderson, A. Ocneanu, D. R.Carlson, A. G. Rowe, & M. Bergés. (2012). BLUED : A Fully Labeled Public Dataset for Event-Based Non-Intrusive Load Monitoring Research. In the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD'12), pp. 1-5.

K. Gröchenig. (2001). Foundations of Time-Frequency Analysis. 10.1007/978-1-4612-0003-1.

K. O'Shea, & R. Nash. (2015). An introduction to convolutional neural networks. arXiv, abs/1511.08458. 10.48550/arXiv.1511.08458.

K. Taunk, S. De, S. Verma, & A. Swetapadma. (2019). A Brief Review of Nearest Neighbor Algorithm for Learning and Classification. In the International Conference on Intelligent Computing and Control Systems (ICCS'19), pp. 1255-1260. 10.1109/ICCS45141.2019.9065747.

L. Breiman. (2001). Random Forests. Machine Learning, vol. 45(1), pp. 5-32. 10.1023/A:1010933404324.

L. de Diego Otón. (2020). Definition of NILM techniques for energy disaggregation in AIIL environments. Available online in: http://hdl.handle.net/10017/43674

L. de Diego-Otón, Á. Hernández Alonso, R. Nieto Capuchino, & M. C. Pérez Rubio. (2021). Evaluación de una Arquitectura CNN para la Identificación de Cargas en NILM. In the XXVIII Seminario Anual De Automática, Electrónica Industrial E Instrumentación (SAAEI'21), pp. 116-121.

L. de Diego-Otón, Á. Hernández Alonso, R. Nieto Capuchino, & M. C. Pérez Rubio. (2022). Comparison of Neural Networks for High-Sampling Rate NILM Scenario. In the IEEE International Symposium on Medical Measurements and Applications Proceedings (MeMeA'22).

M. Bilodeau, & D. Brenner. (1999). Principal components. Theory of Multivariate Statistics (pp. pp. 161-173). Springer. 10.1007/978-0-387-22616-3_10.

M. Gulati, S. Ram, & A. Singh. (2014). An In Depth Study into Using EMI Signatures for Appliance Identification. In the Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient (BuildSys '14), pp. 70-79. 10.1145/2674061.2674070.

M. Kahl, A. Haq, T. Kriechbaumer, & H. A. Jacobsen. (2016). WHITED - A Worldwide Household and Industry Transient Energy Data Set. In the Proceedings of the 3rd International Workshop on Non-Intrusive Load Monitoring, pp. 1-5.

M. Nguyen, S. Alshareef, A. Gilani, & W. G. Morsi. (2015). A novel feature extraction and classification algorithm based on power components using single-point monitoring for NILM. In the IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE'15), pp. 37-40. 10.1109/CCECE.2015.7129156.

M. Ribeiro, L. Pereira, F. Quintal, & N. Nunes. (2016). SustDataED: A Public Dataset for Electric Energy Disaggregation Research. In the Proceedings of ICT for Sustainability, pp. 244–245. 10.2991/ict4s-16.2016.36.

M. Schuster, & K. Paliwal. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, vol. 45(11), pp. 2673-2681. 10.1109/78.650093.

M. Somvanshi, P. Chavan, S. Tambade, & S. V. Shinde. (2016). A review of machine learning techniques using decision tree and support vector machine. In the International Conference on Computing Communication Control and Automation (ICCUBEA'16), pp. 1-7. 10.1109/ICCUBEA.2016.7860040.

M. V. Narkhede, P. P. Bartakke, & M. S. Sutaone. (2022). A review on weight initialization strategies for neural networks. Artificial Intelligence Review, vol. 55(1), pp. 291-322. 10.1007/s10462-021-10033-z.

N. Batra, M. Gulati, A. Singh, & M. B. Srivastava. (2013). It's Different: Insights into home energy consumption in India. In the Proceedings of the 5th ACM Workshop on Embedded Systems for Energy-Efficient Buildings (BuildSys'13), pp.1-8. 10.1145/2528282.2528293.

N. Batra, O. Parson, M. Bergés, A. Singh, & A. Rogers. (2014). A comparison of non-intrusive load monitoring methods for commercial and residential buildings. ArXiv, abs/1408.6595. 10.48550/arXiv.1408.6595.

N. Kehtarnavaz. (2008). CHAPTER 7 - Frequency Domain Processing. In N. Kehtarnavaz (Ed.), Digital Signal Processing System Design (Second Edition) (pp. pp. 175-196). Academic Press. 10.1016/B978-0-12-374490-6.00007-6.

O. Majeed Butt, M. Zulqarnain, & T. Majeed Butt. (2021). Recent advancement in smart grid technology: Future prospects in the electrical power network. Ain Shams Engineering Journal, vol. 12(1), pp. 687-695. 10.1016/j.asej.2020.05.004.

O. Parson, G. Fisher, A. Hersey, N. Batra, J. Kelly, A. Singh, W. Knottenbelt, & A. Rogers. (2015). Dataport and NILMTK: A building data set designed for non-intrusive load monitoring. In the 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 210-214. 10.1109/GlobalSIP.2015.7418187.

P. Adjei, N. S. Sethi, C. P. E. de Souza, & M. A. M. Capretz. (2020). Energy Disaggregation using Multilabel Binarization and Gaussian Naive Bayes Classifier. In the 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON'20), pp. 93-100. 10.1109/UEMCON51285.2020.9298157.

P. Baldi. (2011). Autoencoders, unsupervised learning, and deep architectures. In the Proceedings of the International Conference on Unsupervised and Transfer Learning Workshop (UTLW'11), , vol. 27 pp. 37-50.

P. Bourke. (1996). Cross correlation. Cross Correlation, Auto Correlation—2D Pattern Identification, , n.p.

P. G. Papageorgiou, P. A. Gkaidatzis, G. C. Christoforidis, & A. S. Bouhouras. (2021). Unsupervised NILM Implementation Using Odd Harmonic Currents. In the 56th International Universities Power Engineering Conference (UPEC'21), pp. 1-6. 10.1109/UPEC50034.2021.9548250.

P. Schulman. (1984). Bayes' Theorem—A Review. Cardiology Clinics, vol. 2(3), pp. 319-328. 10.1016/S0733-8651(18)30726-4.

R. Nieto, L. de Diego-Otón, Á. Hernández, & J. Ureña. (2021). Data Collection and Cloud Processing Architecture Applied to NILM Techniques for Independent Living. In the IEEE International Instrumentation and Measurement Technology Conference (I2MTC'21), pp. 1-6. 10.1109/I2MTC50364.2021.9460010.

R. Reed, M. Cox, T. Wrigley, & B. Mellado. (2015). A CPU benchmarking characterization of ARM based processors. Computer Research and Modeling, vol. 7, pp. 581-586. 10.20537/2076-7633-2015-7-3-581-586.

S. Dai, Q. Wang, & F. Meng. (2021). A telehealth framework for dementia care: an ADLs patterns recognition model for patients based on NILM. In the International Joint Conference on Neural Networks (IJCNN'21), pp. 1-8. 10.1109/IJCNN52387.2021.9534058.

S. Hochreiter, & J. Schmidhuber. (1997). Long Short-term Memory. Neural Computation, vol. 9(8), pp. 1735-1780. 10.1162/neco.1997.9.8.1735.

S. Houidi, D. Fourer, & F. Auger. (2020). On the Use of Concentrated Time–Frequency Representations as Input to a Deep Convolutional Neural Network: Application to Non Intrusive Load Monitoring. Entropy, vol. 22(9), sp. 911. 10.3390/e22090911.

S. Katz. (1983). Assessing Self-maintenance: Activities of Daily Living, Mobility, and Instrumental Activities of Daily Living. Journal of the American Geriatrics Society, vol. 31(12), pp. 721-727. 10.1111/j.1532-5415.1983.tb03391.x.

S. Makonin, B. Ellert, I. V. Bajić, & F. Popowich. (2016). Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014. Scientific Data, vol. 3, sp. 160037. 10.1038/sdata.2016.37.

S. Makonin, Z. J. Wang, & C. Tumpach. (2018). RAE: The Rainforest Automation Energy Dataset for Smart Grid Meter Data Analysis. Data, vol. 3(1), sp. 8. 10.3390/data3010008.

S. Na, L. Xumin, & G. Yong. (2010). Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm. In the Third International Symposium on Intelligent Information Technology and Security Informatics, pp. 63-67. 10.1109/IITSI.2010.74.

S. Qi, S. Hou, R. Liu, P. Li, & Y. Zhou. (2021). High-dimensional Feature Optimization Modeling Considering the Characteristics of Non-intrusive Load Identification Model Based on Edge Computing Architecture. In the International Conference on Advanced Electrical Equipment and Reliable Operation (AEERO'21), pp. 1-7. 10.1109/AEERO52475.2021.9708109.

S. Ruder. (2016). An overview of gradient descent optimization algorithms. arXiv, abs/1609.04747. 10.48550/arXiv.1609.04747.

S. Verma, S. Singh, & A. Majumdar. (2021). Multi-label LSTM autoencoder for non-intrusive appliance load monitoring. Electric Power Systems Research, vol. 199, sp. 107414. 10.1016/j.epsr.2021.107414.

S. Yaemprayoon, & J. Srinonchat. (2022). Non-Intrusive Load Monitoring using Multi-Layer Perceptron for Appliances Classification. In the 19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON'22), pp. 1-4. 10.1109/ECTI-CON54298.2022.9795518.

T. Kriechbaumer, & H. Jacobsen. (2018). BLOND, a building-level office environment dataset of typical electrical appliances. Scientific Data, vol. 5(1), sp. 180048. 10.1038/sdata.2018.48.

T. Le, J. Kim, & H. Kim. (2016). Classification performance using gated recurrent unit recurrent neural network on energy disaggregation. In the International Conference on Machine Learning and Cybernetics (ICMLC'16), pp. 105-101. 10. 10.1109/ICMLC.2016.7860885.

T. Picon, M. Nait-Meziane, P. Ravier, G. Lamarque, C. Novello, J. C. Le Bunetel, & Y. Raingeaud. (2016). COOLL: Controlled On/Off Loads Library, a Public Dataset of High-Sampled Electrical Signals for Appliance Identification. arXiv, abs/1611.05803. 10.48550/arXiv.1611.05803.

T. -T. -H. Le, H. Kang, & H. Kim. (2020). Household Appliance Classification Using Lower Odd-Numbered Harmonics and the Bagging Decision Tree. 10.1109/ACCESS.2020.2981969.

V. Cherkassky, & F. Mulier. (1999). Vapnik-Chervonenkis (VC) learning theory and its applications. IEEE Transactions on Neural Networks, vol. 10(5), pp. 985-987.

Wu, X., Gao, Y., & Jiao, D. (2019). Multi-Label Classification Based on Random Forest Algorithm for Non-Intrusive Load Monitoring System. Processes, vol. 7(6), sp. 377. 10.3390/pr7060337.

X. Glorot, & Y. Bengio. (2010). Understanding the difficulty of training deep feedforward neural networks. In the Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, vol. 9, pp. 249-256.

X. He, H. Dong, W. Yang, & J. Hong. (2022). A Novel Denoising Auto-Encoder-Based Approach for Non-Intrusive Residential Load Monitoring. Energies, vol. 15(6), sp. 2290. 10.3390/en15062290.

X. Li, S. Wang, & Y. Cai. (2019). Tutorial: Complexity analysis of singular value decomposition and its variants. arXiv, abs/1906.12085. 10.48550/arXiv.1906.12085.

Y. Bengio, A. Courville, & P. Vincent. (2013). Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35(8), pp. 1798-1828.

Y. H. Lin. (2022). An advanced smart home energy management system considering identification of ADLs based on non-intrusive load monitoring. Electrical Engineering, , n.p. 10.1007/s00202-022-01546-z.

Y. Himeur, A. Alsalemi, F. Bensaali, A. Amira, & A. Al-Kababji. (2022). Recent trends of smart nonintrusive load monitoring in buildings: A review, open challenges, and future directions. International Journal of Intelligent Systems, , pp. 1-56. 10.1002/int.22876.

Y. Lin, & M. Tsai. (2011). Applications of hierarchical support vector machines for identifying load operation in nonintrusive load monitoring systems. In the 9th World Congress on Intelligent Control and Automation, pp. 688-693. 10.1109/WCICA.2011.5970603.

Z. Xiao, W. Gang, J. Yuan, Y. Zhang, & C. Fan. (2021). Cooling load disaggregation using a NILM method based on random forest for smart buildings. Sustainable Cities and Society, vol. 74, sp. 103202. 10.1016/j.scs.2021.103202.

Appendix A Budget

This chapter will describe the theoretical cost of the whole proposal. These accounts include material costs of the software and hardware tools and professional fees. All this information is presented in the following tables (from Tables 10 to 12).

A.1 Material cost

This section details the cost of the different materials used (including VAT).

Table 10. Material costs.

ITEM		CONCEPT	AMORTIZATION	UNIT COST	TIME	TOTAL COST
Hardware	***Window PC i7 3.6 GHz***	Desktop laboratory PC	4 years (26 % lineal)	390,00 €/year	0,75 year	292,50 €
Software	***MATLAB© R2020b***	Signal processing	-	489,00[2] €/year	0,75 year	366,75 €
	Python 3.8.3	Deep Learning design	-	0,00 €	-	0,00 €
	Windows 10 Pro	Operating System	-	259,00[2] €/year	0,75 year	194,25 €
	Office 365	Document preparation	-	134,16[2] €/month	0,125 year	16,77 €
MATERIAL TOTAL COST						870,27 €

A.2 Professional fees

In this section, the different professional fees are calculated as gross income (excluding VAT). These include all the professional activities related to the project where an industrial profit of around 6 % has already been taken into account.

Table 11. Professional fees.

TASK	COST	TIME	TOTAL COST
Junior engineering design	60,00 €/hour	720 hours	43.200,00 €
Typing	25,00 €/hour	240 hours	6.000,00 €
PROFESSIONAL FEES TOTAL COST			49.200,00 €

[2] Standard annual license price. This product provides the right to use the software for a fixed duration of one year.

A.3 Total costs

The budget calculations for this Master's Thesis end adding the total costs of materials and the total costs of professional fees, including other costs associated with the application of VAT, stipulated at 21 %, based on the budget of execution by contract.

Table 12. Total costs.

CONCEPT	COST
Material total cost	870,27 €
Professional fees total cost	49.200,00 €
VAT of professional fees (21%)	10.332,00 €
TOTAL COST	60.402,27 €

The final estimated budget for the completion of this Master's Thesis amounts to a total of 60.402,27 € (sixty thousand four hundred and two euros and twenty-seven cents).

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR