University of Stavanger

KONRAD KRZYSZTOF JARCZYK
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Identification of Electrical Devices Using Machine Learning: A Study in Collaboration with Origin AS

Master's Thesis - Computer Science - June 2024

I, **Konrad Krzysztof Jarczyk**, declare that this thesis titled, “Identification of Electrical Devices Using Machine Learning: A Study in Collaboration with Origin AS” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a master’s degree at the University of Stavanger.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Preface

This thesis has undergone several modifications based on feedback received during the poster presentation. Notably, the title has been refined to better reflect the scope and objectives of the research. Additionally, a link to the GitHub repository containing the code and detailed instructions has been included at the beginning of the Approach chapter 3.

Furthermore, I would like to acknowledge the use of ChatGPT in this thesis. OpenAI's ChatGPT (June 2024 version) was used to transform text and sentence structures. The tool provided suggestions for rephrasing and restructuring sentences, improving clarity, coherence, and readability. It did not affect the scientific data, outcomes, or the methodology used in this thesis [1].

Additionally, the image on the front page was generated using AI, specifically through OpenAI's DALL-E model, to visually represent the concept of this research [1].

Abstract

This thesis aimed to develop reliable methods for identifying electrical devices in a household based on aggregate electrical signals from a main electric monitor. The study involved creating or obtaining a suitable dataset, implementing the NILM procedure, utilizing machine learning methods to determine their effectiveness in learning device usage patterns, and creating a practical application using these models. Random Forest, RNN/LSTM, and CNN models were employed, with the Random Forest model achieving the highest overall accuracy. The RNN/LSTM model showed strong potential in capturing temporal dependencies. Conversely, the CNN model did not perform well, indicating that further research is needed to optimize its application to NILM tasks. The study highlighted the importance of using comprehensive datasets that include multiple devices simultaneously, allowing the models to learn interactions and overlapping usage patterns. While the development of a practical application was initiated, it was not fully realized. Despite some limitations, such as the lack of adaptability testing across different households, the findings provide a foundation for future studies and practical applications in household energy management.

Acknowledgements

I would like to thank my supervisor, Nejm, for their enthusiasm and invaluable assistance in writing this thesis. I also want to express my gratitude to my girlfriend, Ida, for her constant support during stressful times and for always being there for me. Additionally, I am deeply thankful to my mum and dad for their unconditional love and care. Lastly, I want to extend my thanks to Rune, my favorite boss, for believing in me.

Contents

Nomenclature

- **Active Power (P)**: The rate at which electrical energy is converted to work, measured in watts (W). It represents the actual power consumed by electrical devices.
- **Reactive Power (Q)**: The power that oscillates between the source and the load, necessary for the operation of magnetic fields in motors and inductors. It does not perform any work and is measured in volt-amperes reactive (VAR).
- **Apparent Power (S)**: Represents the total power in an electrical circuit, combining both active and reactive power, measured in volt-amperes (VA).
- **Power Factor (PF)**: The ratio of active power to apparent power, indicating the efficiency of power usage.
- **Phase Angle (φ)**: The phase difference between the voltage and current waveforms, indicative of how the voltage and current waves are synchronized.
- **Voltage (V) and Current (I)**: Fundamental electrical parameters indicating the electrical force and flow of electric charge, respectively.
- **Frequency (f)**: The frequency of the AC power supply, measured in hertz (Hz), which indicates the rate at which the current changes direction per second.

Chapter 1

Introduction

1.1 Background and Motivation

In contemporary society, the concept of ”smart homes” has evolved from a futuristic idea depicted in science fiction to a practical reality in everyday life. These homes are equipped with systems that allow for the remote control and automation of lighting, temperature, and security, symbolizing a modern approach to home management. Users interact with these systems through smartphone apps, enhancing convenience and control. Smart devices can significantly improve energy efficiency, security, and overall quality of life.

Despite these advantages, not every household is equipped with smart devices. Many homes still lack the infrastructure needed to monitor and control energy consumption effectively. This gap means that a significant portion of the population cannot benefit from the potential savings and efficiencies that smart home technologies offer. The challenge is particularly pronounced in older homes where retrofitting with smart technology can be expensive and complex.

Furthermore, the proliferation of various electrical devices in homes presents significant challenges in energy management. Homeowners often lack precise, easily accessible information about how, when, and where electricity is used in their homes. This lack of visibility can lead to inefficiencies and higher energy costs, which is a growing concern as energy prices rise and ecological sustainability becomes a priority.

This thesis addresses these challenges by exploring Non-Intrusive Load Monitoring (NILM) technologies. NILM offers a sophisticated method for analyzing

energy consumption patterns without the need for individual appliance metering. By analyzing energy usage data, NILM systems can identify which appliances are operational and estimate their energy consumption, all without direct interference in household activities. This technology not only aids in effective energy management but also aligns with the evolving landscape of smart homes by providing critical insights into energy usage patterns.

The contributions of this thesis are threefold:

- **Optimizing Electricity Usage**: By providing detailed insights into the energy consumption patterns of individual devices, this research enables homeowners to optimize their electricity usage, potentially leading to significant cost savings.
- **Detecting Unsupposed Device Usage**: The methods developed can detect unusual or unintended use of devices, such as appliances left on accidentally. This feature can enhance safety by preventing potential hazards.
- **Improving Security and Health**: By monitoring and analyzing device usage patterns, the system can alert homeowners to irregularities that may indicate security breaches or health risks, such as forgetting to turn off the oven or detecting malfunctioning devices.

1.2 Objectives

The primary aim of this study is to develop and refine a system for Non-Intrusive Load Monitoring (NILM), which is the process of identifying individual electrical devices within a household based on aggregated power input data. This involves several sub-objectives:

- **Dataset Acquisition**: To collect or obtain a comprehensive dataset that captures both the aggregated power usage and the individual consumption of all devices within the network. This detailed data is essential to effectively train models and serve as the foundation for all subsequent analyses.
- **Literature Review**: To conduct a thorough review of existing literature on NILM technologies to understand current methods, challenges, and advancements in the field.

- **Following the NILM Procedure**: To implement the standard NILM procedure, which includes preprocessing data, feature extraction, event detection, and load disaggregation. This ensures a structured approach to analyzing energy consumption data.
- **Algorithm Development and Implementation**: To explore and implement various machine learning algorithms, comparing traditional statistical techniques with modern machine learning approaches to determine the most effective methods for disaggregating household energy consumption into device-specific usage.
- **Evaluation of Results**: To analyze and interpret the results obtained from the NILM systems, identifying patterns and insights that can contribute to more energy-efficient usage behaviors in households.
- **Application Development**: To develop a user-friendly interface that leverages the created models to provide real-time insights into household energy management. This application will help users monitor energy consumption, detect anomalies, and optimize power usage within their homes.

The outcomes of these objectives will contribute to the field of smart home technology by providing improved tools for energy management, ultimately aiding homeowners in better understanding and reducing their energy consumption.

1.3 Outline

This thesis is organized into several chapters, each designed to systematically address different aspects of Non-Intrusive Load Monitoring (NILM). The structure is as follows:

- **Chapter 1: Introduction** This chapter introduces the topic of NILM, outlining the motivations and objectives of the research. It provides a brief overview of the challenges and the importance of energy management in smart homes.
- **Chapter 2: Background** This chapter provides a detailed exploration of the key concepts and technologies related to NILM. It includes a review of

existing research in the field, discussing both the achievements and limitations. This background serves as the foundation for the subsequent development of techniques addressed in this thesis.

- **Chapter 3: Approach** In this chapter, the specific methodologies employed in the research are detailed. It outlines the approach taken for following the NILM procedure. This includes a description of the machine learning models tested and the rationale behind their selection.
- **Chapter 4: Experimental Evaluation** This chapter presents the findings of the study. It describes the outcomes of the applied methodologies and algorithms, providing a detailed analysis of the data. The results are critically assessed to determine the efficacy of different NILM techniques in various scenarios.
- **Chapter 5: Discussion** The discussion chapter interprets the results, linking them back to the research questions and the existing literature discussed in Chapter 2. This section evaluates the implications of the findings, discusses potential limitations of the study, and suggests areas for further research.
- **Chapter 6: Conclusions** The concluding chapter summarizes the key findings and contributions of the thesis.

Chapter 2

Background

Chapter Background aims to explain concepts related to research objectives. It also discusses background theory, methodologies and limitations.

2.1 Introduction to NILM

The concept of Non-Intrusive Load Monitoring (NILM) is thoroughly explained in the article ”Overview of non-intrusive load monitoring and identification techniques” [2]. This article provides an extensive review of the methods used for monitoring and identifying electrical energy consumption and the operational condition of individual devices. It distinguishes between ”intrusive” and ”non-intrusive” systems, highlighting that intrusive load monitoring (ILM) requires physical sensors installed at each appliance, whereas non-intrusive methods monitor energy usage externally without direct interaction with the appliances.

The two primary techniques of load monitoring are:

- **Intrusive Load Monitoring (ILM)**: This technique involves installing sensors at each appliance, providing highly accurate power usage details. However, it is expensive and complex due to the extensive equipment and wiring required.
- **Non-Intrusive Load Monitoring (NILM)**: This method uses a single meter installed at the electrical service entrance to monitor the entire household’s energy consumption. NILM identifies electrical appliances based on

their unique electrical signatures from this aggregated data. It is more cost-effective and convenient than ILM but can be complex in terms of data analysis and may be less accurate.

Both methods have their challenges and advantages, but the focus here is on NILM. NILM techniques involve analyzing power measurements to assign unique electrical signatures to various appliances. Additionally, recognizing state patterns, where appliances like dishwashers or refrigerators operate in specific cycles, aids in identification. For example, a stove burner might heat up by cycling on and off, or a blender might consume more power initially before stabilizing. Figure 2.1 illustrates how electrical patterns can be classified to their respective devices.

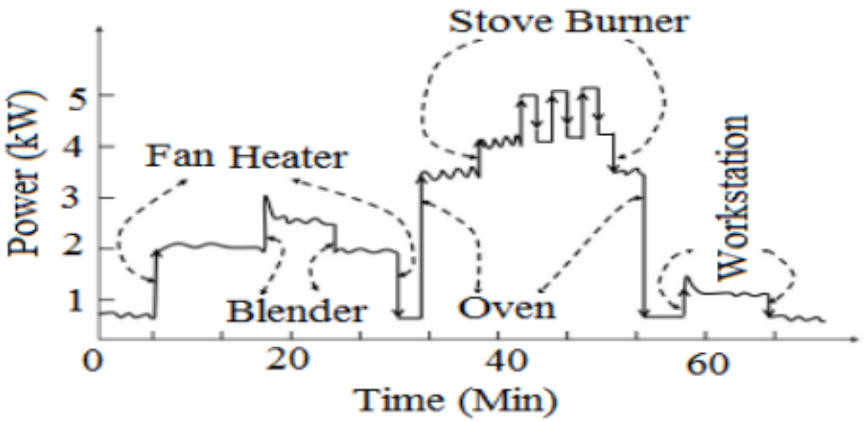


Figure 2.1: Electrical patterns of different devices [2].

In summary, NILM techniques offer several advantages, including lower cost, easier installation, and a less invasive nature, making them more acceptable to consumers and easier to implement in many households. Despite these benefits, NILM systems can be less accurate than their intrusive counterparts and may struggle to distinguish between devices with identical electrical signatures or those operating at constant low power levels. Nevertheless, NILM's ability to provide detailed insights into appliance-specific energy usage is invaluable, enabling utilities and consumers to understand energy consumption patterns more effectively.

2.2 Historical Context of NILM

The development and evolution of Non-Intrusive Load Monitoring (NILM) technologies began with foundational work in the late 20th century. One of the pioneering figures in this field is George W. Hart, whose early work at the Massachusetts Institute of Technology laid the groundwork for NILM by introducing the concept of disaggregating energy usage from a single measurement point [3].

Developments in NILM Technology

1. **Initial Conceptualization (1980s):** George W. Hart developed algorithms capable of detecting changes in total power demand to infer the usage of individual appliances. This disaggregation is based on detecting distinctive changes or 'events' in the voltage and current, which indicate appliance switches turning on or off [3]. This foundational work led to the initial practical applications of NILM in energy management systems, allowing for the monitoring and optimization of energy consumption in industrial and residential settings. Hart's methodology set the stage for further research and development in NILM, emphasizing the potential for detailed energy use analysis without intrusive hardware installations.

2. **Integration of Machine Learning (2000s onwards):** The integration of machine learning techniques marked a significant advancement in NILM technology. Researchers began using methods such as Hidden Markov Models (HMM) [4], Support Vector Machines (SVM) [5], and deep learning approaches [6] to improve disaggregation performance. These techniques significantly enhanced the accuracy and robustness of NILM systems, making it possible to handle a wider variety of appliances and more complex usage patterns. The application of machine learning has enabled NILM systems to adapt to different households and appliances, improving their practical utility and paving the way for widespread adoption in smart grid environments [7].

3. **Recent Advances and Smart Grid Integration (2010s-present):** The proliferation of smart meters and the Internet of Things (IoT) has significantly enhanced NILM capabilities, enabling real-time feedback and integration with smart home management systems [8]. These advancements

promote energy efficiency and consumer engagement by providing detailed, actionable insights into energy consumption patterns. Furthermore, the integration with smart grid technologies has enabled more dynamic and responsive energy management, supporting the broader goals of sustainability and grid stability [9].

2.3 Problem Formulation

The mathematical formulation of Non-Intrusive Load Monitoring (NILM) based on an active power objective can be succinctly described by the equation:

$$x(t) = \sum_{i=1}^{N} y_i(t) + \epsilon(t)$$

where $x(t)$ represents the total power consumption at time t measured at the household's main electrical meter, $y_i(t)$ denotes the power consumed by the i-th appliance at time t, N is the total number of appliances, and $\epsilon(t)$ represents the noise, measurement errors, and unidentified loads.

In NILM research, a critical aspect is how appliance data are classified, particularly in how appliances are recognized as being either 'on' or 'off' or by quantifying the exact energy they are consuming. While classifying appliances simply as 'on' or 'off' is more common due to its simplicity, some studies propose more granular energy consumption measurements for enhanced accuracy and insight, although these are less prevalent due to increased complexity and data requirements.

Typically, the identification of appliance states can be framed either as a classification problem, where each operating state of a device is a separate class, or as a regression problem, where the exact power usage of a device is predicted. Classification approaches allow for the identification of distinct states, such as 'on', 'off', and different power levels, effectively clustering these states into predefined categories. Regression approaches, on the other hand, aim to predict the exact value of power consumption, providing more detailed information about the device's energy usage [10].

While regression approaches can provide more precise energy monitoring by giving exact power usage values, they tend to be more complex. This complexity

arises because regression models must capture continuous variations in power consumption, which often requires more advanced modeling techniques and greater computational resources. Additionally, regression models typically require more detailed and high-resolution data to achieve accurate predictions.

Figure 2.2 illustrates the difference between classification and regression approaches in NILM. In the classification approach, the appliance states are clustered into predefined categories, such as 'on', 'off', and various power levels. In the regression approach, the exact power usage values are predicted, providing a continuous output that represents the actual energy consumption of the appliance.

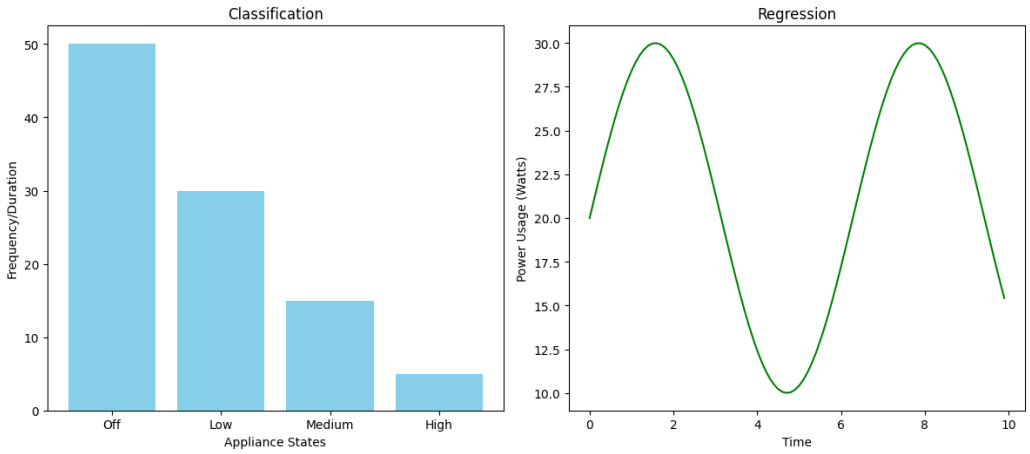


Figure 2.2: Comparison of Classification and Regression Approaches in NILM.

The general steps involved in NILM include data acquisition, signal processing, feature extraction, event detection, and load disaggregation, where each step builds upon the previous to create an effective monitoring system [11].

2.3.1 Steps in NILM

1. **Data acquisition**: In first step energy consumption data is collected from a smart meter. This data is typically sampled at regular intervals, typical ranging from microseconds to seconds.

2. **Signal processing**: is a technique to clean data from any sort of noise or corrupted/incorrectly formatted data and prepare it for further steps. Data

normalization is one of these processes.

3. **Feature extraction**: aims to distinguish and identify relevant features between different usage patterns. Common features include changes in power levels, consumption over time or other information, that helps in differentiating devices.

4. **Event detection**: focuses on identifying changes in consumption patterns, that can tell if a given device is on or off.

5. **Load disaggregation**: Involves breaking down the aggregate energy signal into individual components attributable to each appliance. This step uses various algorithms to identify and separate the energy consumption patterns of different devices within the household.

2.4 Smart Meters and Data Acquisition

Smart meters are crucial components in modern energy management systems, enabling detailed data collection and communication between electric utilities and consumers. This section explores their role in data acquisition and their significance in Non-Intrusive Load Monitoring.

These advanced metering devices measure electric energy consumption at high temporal resolutions, such as every second, 30 seconds, 5 minutes, or hourly intervals. Smart meters facilitate two-way communication, enabling not only the remote reporting of energy usage but also supporting dynamic pricing and demand response programs. They are instrumental in collecting diverse types of data, as listed in Nomenclature, which enhance energy management and customer interaction [12].

In NILM systems, smart meters collect data from various devices, allowing algorithms to learn usage patterns based on the gathered information. Figure 2.3 illustrates the data flow from devices to smart meters and how responses are communicated back to the devices. This flow includes data collection from various appliances, secure transmission to utility data centers, processing and analysis, and the communication of insights and commands back to the consumer devices for optimized energy management.

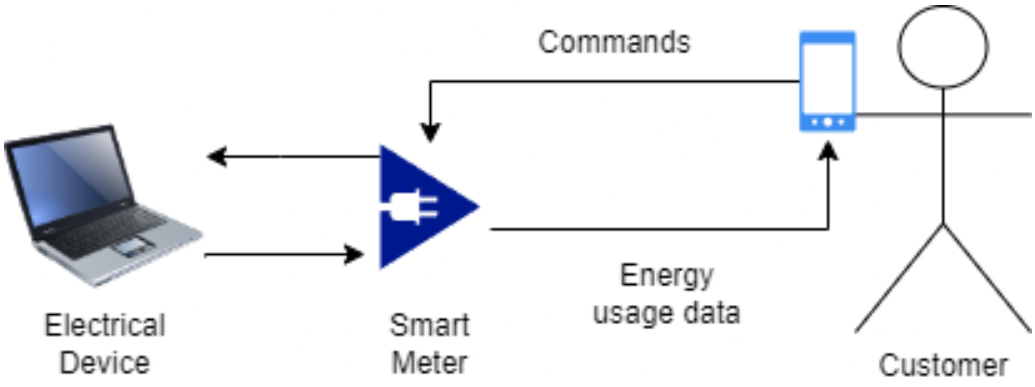


Figure 2.3: Data flow from smart meters to customer devices and back.

2.5 Electrical Devices and Usage Patterns

Electrical devices in households can be categorized by their operational states, a division that is crucial for enhancing NILM systems. These devices typically fall into three categories based on the number of operational states they exhibit:

Two-State Appliances (ON/OFF): This category includes devices such as kettles, non-dimmable light bulbs, simple heaters, and toasters. These appliances toggle between an ON state, where they consume a specific amount of power (e.g., a kettle using 3000 W), and an OFF state, where no electricity is consumed. Identification of such devices using NILM techniques is generally straightforward due to their distinct and consistent power signatures, which differ markedly between the ON and OFF states. An example of such a device is illustrated in Figure 2.4,

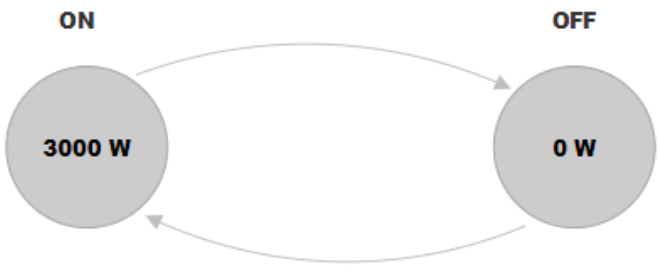


Figure 2.4: Example of a two-state device.

showing the binary operation of toggling between ON and OFF.

Multi-State Appliances: These appliances have a fixed number of operating states beyond just ON and OFF. Each state corresponds to a different mode of operation. For instance, a washing machine (as shown in Figure 2.5) may have separate modes for washing, rinsing, and spinning. These modes follow a specific sequence—filling with water, heating the water, and then agitating the clothes—which creates distinct and identifiable power consumption patterns, making them more challenging but still feasible for NILM systems to identify due to their repetitive and predictable nature. Other examples include dishwashers, coffee makers, and vacuum cleaners that have settings like low and high power.

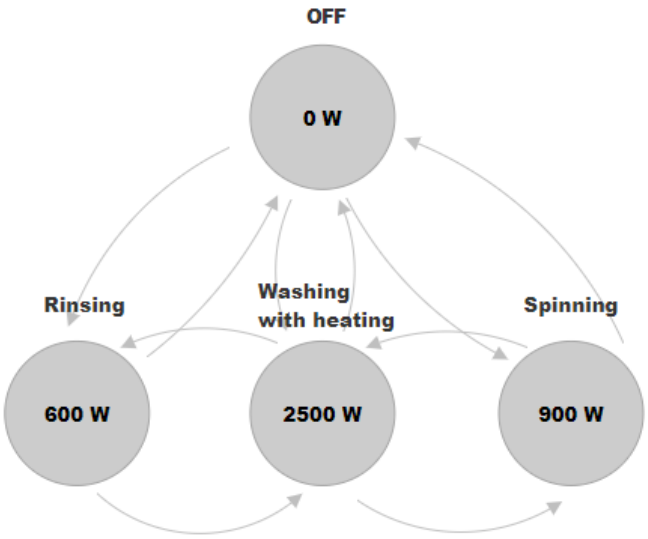


Figure 2.5: Example of a multi-state device.

Variable-State Devices These devices, such as dimmable lights, do not have fixed operational states but instead operate across a continuum of settings. For example, a dimmable light bulb can adjust its brightness, and consequently its power consumption, anywhere between 25 W and 250 W based on user preference, as depicted in Figure 2.6. Other variable-state devices include power drills, treadmills, and electric ovens. The flexibility in their operation means they do not follow specific patterns, making them difficult to identify consistently within NILM systems.

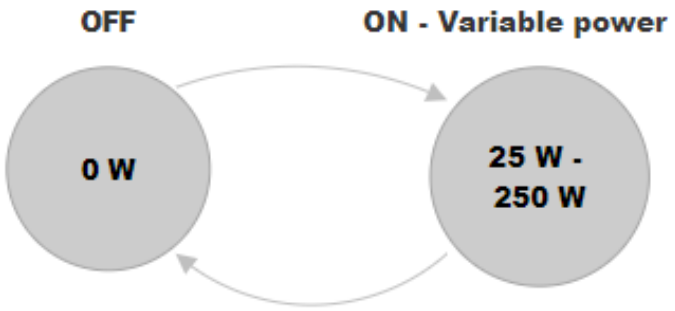


Figure 2.6: Example of a variable state device.

Each type of appliance presents unique challenges and opportunities for NILM systems. Understanding the specific operational states and their associated power consumption patterns is essential for accurately identifying and monitoring these devices, thereby improving the overall efficiency and effectiveness of NILM [13].

2.6 K-means Clustering Technique

K-means clustering is a popular and straightforward unsupervised machine learning algorithm used to solve clustering problems. The algorithm partitions a set of data points into K distinct non-overlapping subsets or clusters. The objective is to minimize the total intra-cluster variance, or the squared error between the points and their respective cluster centroid [14, 15].

In the context of NILM, k-means clustering can be highly beneficial for identifying operational states of household appliances from their energy consumption data. For instance, consider a device with two operating states: 'on' and 'off'. The 'on' state may have power consumption values around 1000 Watts with a variance of ±100 Watts, while the 'off' state (or standby) may have values up to 100 Watts.

Applying k-means clustering helps in grouping these states effectively. By clustering the power consumption data, we can categorize these measurements into distinct states. This preprocessing step simplifies the classification problem for NILM by providing clear clusters of 'on' and 'off' states, making it easier for

machine learning algorithms to train on these grouped values.

Figure 2.7 illustrates how k-means clustering can group power consumption values into distinct states, aiding in the classification process.

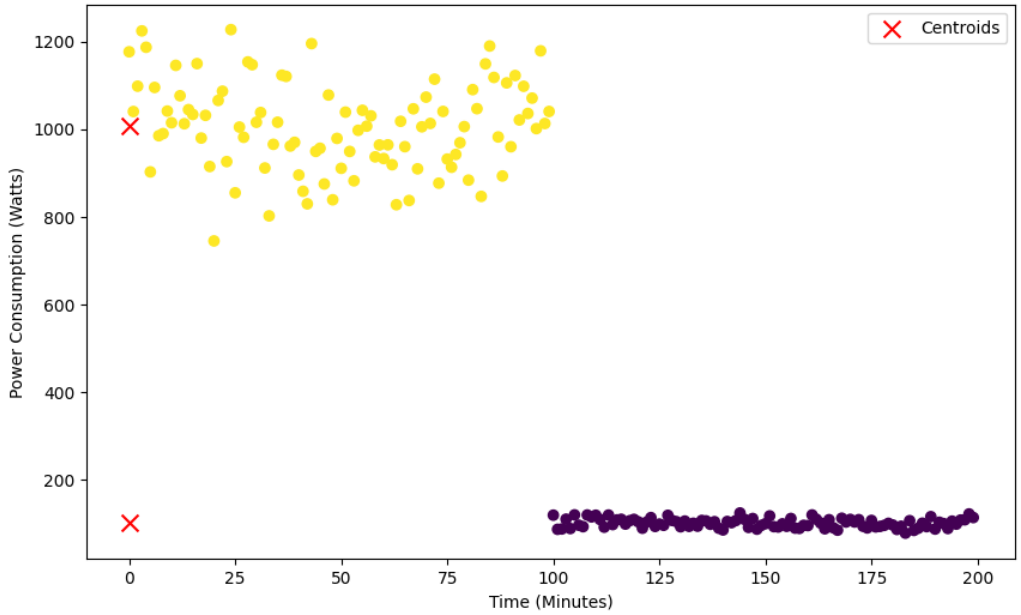


Figure 2.7: Example of k-means clustering applied to NILM data, grouping 'on' and 'off' states of an appliance.

2.7 Load Disaggregation Techniques

In this section, various load disaggregation methods are briefly presented.

2.7.1 Hidden Markov Model

Hidden Markov Models (HMMs) are statistical models which assume that the system being modeled follows a Markov process with unobserved, hidden states. In Non-Intrusive Load Monitoring (NILM), HMMs model the transition probabilities between different appliance states, which are not directly observable but whose outputs are dependent on these states. Each state has a probabilistic distribution over possible output tokens, which allows the sequence of outputs to reveal the sequence of states that the system transitions through. These models

are particularly effective in NILM for modeling the stochastic nature of appliance usage, where both the transition probabilities and the output probabilities adhere to the Markov property—being dependent only on the current state and independent of previous states and outputs [13].

2.7.2 Random Forest

Random Forest is a sophisticated ensemble learning technique that utilizes multiple decision trees to improve predictive accuracy and control over-fitting. Each tree in the forest predicts independently and contributes to the final decision, making it robust against noise and variance in large datasets. In Non-Intrusive Load Monitoring (NILM), Random Forest excels in disaggregating appliance load by analyzing patterns of electricity usage across various states such as 'on' and 'off'. This technique effectively handles the complexity inherent in NILM due to its ability to model non-linear relationships without prior assumptions about data distribution. The integration of Random Forest in NILM has been explored for its efficiency in classifying multi-state appliances and enhancing energy estimation by employing techniques like integer linear programming for precise power breakdown [16].

2.7.3 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are designed to handle sequential data, making them ideal for applications that require learning from data where order matters, such as time-series analysis. In Non-Intrusive Load Monitoring (NILM), RNNs are utilized to model and predict energy load profiles over time, capturing the dynamic changes in energy usage [12].

Long-Short Term Memory Networks

Long-Short Term Memory networks (LSTMs) are an advanced type of RNNs that can learn long-term dependencies in data, a crucial feature for NILM applications where appliance usage patterns vary significantly over time. LSTMs effectively manage the challenges posed by sequence length and time lags in energy consumption data [12].

Sequence To Sequence LSTM

Sequence to Sequence (Seq2Seq) LSTM models, originally developed for tasks like machine translation, are adept at handling complex sequences where both input and output are sequences [12].

2.7.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of deep learning model primarily renowned for image processing tasks such as recognition and classification. These networks inherently identify key features from the data, thereby minimizing the requirement for manual feature engineering. In the context of Non-Intrusive Load Monitoring (NILM), CNNs treat aggregate power consumption waveforms similarly to images, enabling them to learn and identify distinctive power usage patterns of various appliances [12].

2.8 Process Mining

An innovative approach to Non-Intrusive Load Monitoring (NILM) that goes beyond traditional machine learning techniques is process mining. Instead of relying solely on aggregated power consumption data to identify devices, process mining focuses on the sequence of device operations within a network. This method considers the order in which devices turn on, their power usage levels (high, low or medium), and other operational patterns.

For instance, in the morning, a single male living in a household might follow a routine where he first turns on the coffee machine, then the toaster, runs the dishwasher, and finally takes a shower, as illustrated in Figure 2.8. Recognizing such household habits can significantly enhance the accuracy of device identification, especially when combined with machine learning techniques. While machine learning can identify that a morning energy peak is due to a kettle, process mining can further predict subsequent devices likely to be used. Thus, process mining serves as a powerful tool to improve NILM by providing a sequential and contextual understanding of device usage.

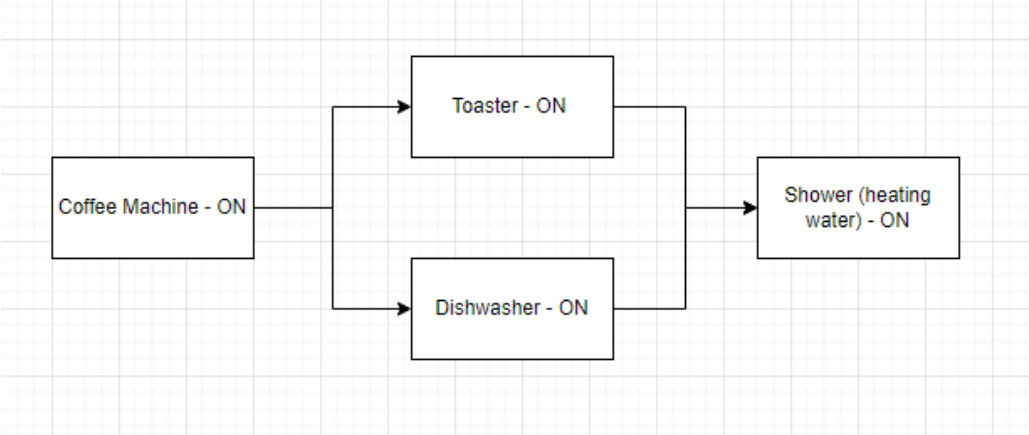


Figure 2.8: Sequence of events.

2.8.1 Process mining algorithms

Process mining algorithms apply mathematical rules to derive process models using data mining techniques. These algorithms are widely used to identify bottlenecks, improve efficiencies, and understand the states of business processes in a data-driven manner. The most popular process mining algorithms include Alpha Miner, Heuristic Miner, Inductive Miner, and Genetic Miner. This work focuses on the Alpha Miner algorithm.

Alpha Miner utilizes event logs as its primary data source. Then transforms event logs into models that represent direct-follow, sequence, parallel, and choice relations. In simpler terms, Alpha Miner creates a timestamped process model flow, similar to the sequence shown in Figure 2.8 [17].

2.9 Review on existing NILM approaches

In this section some existing NILM approaches are discussed.

2.9.1 Article 1: Real-Time Identification of Electrical Devices

Article from 2016 titled ”Real Time Identification of Electrical Devices through Power Consumption Pattern Detection” [18] gives the feasibility of identifying electrical devices in real time by analyzing their unique power consumption pattern. Author of this article - Abeykoon claims, that including factors as active

power, reactive power, phase shift, and the root mean square (RMS) values of voltage and current across different operating modes of the devices, give at most 98 % of accuracy in real time identification. The study posits that each electrical device possesses a distinctive electrical signature, which can be decoded from collected parameters. With testing with different machine learning techniques such as Support Vector Machines (SVM), Artificial Neural Networks (ANN), K-Means Classification, Silhouette Classification, and Mean-Shift Classification, it is possible to classify these signatures with high accuracy.

However, the article provides limited information on the methodology for data collection and training information, which raises questions about the diversity and representativeness of the dataset used for model training and validation. In synthesizing the contributions of Abeykoon et al.'s study, it is evident that the real-time identification of electrical devices through power consumption pattern detection represents a significant leap forward in NILM research.

2.9.2 Article 2: Neural NILM: Deep Neural Networks Applied to Energy Disaggregation

The study "Neural NILM: Deep Neural Networks Applied to Energy Disaggregation" [19] by Jack Kelly and William Knottenbelt investigates energy disaggregation using three distinct neural network architectures: Long Short-Term Memory (LSTM) networks, denoising autoencoders, and a regression-based network designed to predict appliance activation profiles. Conducted on the UK-DALE dataset, the research focuses on frequently used appliances like kettles, fridges, washing machines, microwaves, and dishwashers, each with networks trained to recognize their distinct energy consumption patterns.

The researchers adopted a windowing method to manage data segments specific to each appliance's typical usage duration—for instance, 13 minutes for kettles and 2.5 hours for dishwashers. A separate model was trained for each device, using different window sizes tailored to the respective appliance's usage patterns.

Results from neural networks analysis were then compared to traditional methods such as combinatorial optimization and hidden Markov models (HMM), with performance assessed based on the accuracy of disaggregation.

Results highlighted that the denoising autoencoder and the regression-based network significantly outperformed traditional methods, with accuracy scores of

0.93 and 0.97, respectively, compared to 0.79 for combinatorial optimization and 0.70 for HMM. The LSTM network, however, scored lower at 0.66, indicating some challenges and explained by authors as possible problem in handling appliances with multi-state operations.

2.9.3 Nilmtk

Nilmtk is an open-source library for load disaggregation [20]. It was developed in 2014 by three colleagues to provide a standardized toolkit for NILM research. Nilmtk has been widely referenced in numerous publications, highlighting its significance in the field and the robustness of the methods it presents. The initial plan for this thesis included comparing the methods developed herein with those implemented in Nilmtk to evaluate their performance and accuracy.

However, due to the age of the package and compatibility issues with current software environments, it was impossible to run and test Nilmtk. This limitation hindered the direct comparison between the new methods presented in this thesis and the established methods within Nilmtk.

2.9.4 Tibber's Approach

Tibber [21], a Scandinavian company, offers innovative solutions for energy optimization using both software and hardware. Their platform facilitates the intelligent use of electricity by timing activities such as car charging sessions to coincide with periods of low electricity prices. Tibber focuses on providing users with real-time data on overall power consumption.

Although Tibber employs NILM solutions, the specifics of their approach are not accessible. However, the idea behind gathering data that can be used to train individual patterns of usage within households is valuable. Through their app, users can observe the impact of turning various devices on and off, allowing them to manually identify and manage high-energy-consuming appliances. This user-involved approach helps optimize energy use without the need for specific appliance identification technology like NILM.

Tibber's method serves as a solid foundation for collecting data without the need for intrusive smart meters, capturing how different devices influence the total power output. A similar system could be interesting to investigate further,

where the collected data could serve as training data for machine learning methods.

2.10 Challenges and Complexity in NILM

Non-Intrusive Load Monitoring faces significant challenges due to the varied nature of electrical devices and their consumption patterns. The challenge lies in recognizing its energy signature among overlapping signals from other devices. For instance, consider a morning routine where a coffee maker and a toaster is switched on at the same time and both use similar energy. These consecutive usages can blur the distinct electrical signatures, complicating the disaggregation process.

Appliances often depend on each other's operation cycles, influenced by user habits or programmed settings. This interdependence and sequences of operation add layers of complexity to energy monitoring.

Viewing household dynamics as a living process illustrates the challenge in NILM. Household energy use is not static; it varies throughout the day, influenced by human behavior, weather conditions, and even real-time energy prices. Each variable can significantly impact the overall energy consumption pattern.

For example, the use of heating systems may increase significantly during cold weather, coinciding with more frequent use of hot water, which can challenge the NILM system to accurately attribute increased energy usage to the correct sources. Moreover, the influence of real-time electricity pricing can lead to strategic usage of certain appliances during off-peak hours, adding another variable that NILM systems must account for.

2.11 Research Tools

In this section, we introduce the programming language and the necessary libraries used for this research.

2.11.1 Python for NILM

Python is the preferred programming language for NILM research due to its diversity, ease of use, and the extensive scientific libraries. Python's syntax is straight-

forward, making it accessible for both new and experienced programmers. Additionally, Python is highly extensible and supports integration with other languages and tools, which is crucial for complex NILM tasks that might require diverse computational approaches [22].

Python has very powerful libraries for data manipulation, machine learning, and signal processing. These libraries provide the tools needed to preprocess data, build and evaluate models, and perform detailed analyses of power consumption patterns.

2.11.2 Relevant Libraries

Several Python libraries are particularly relevant in this NILM research:

- **Pandas** [23]: Pandas is a fast, powerful, and flexible open-source data analysis and manipulation library. It provides data structures like DataFrames, which are essential for handling time series data.
- **Scikit-learn** [24]: Scikit-learn is a widely used machine learning library in Python. It offers simple and efficient tools for data mining and data analysis, including classification, regression, clustering algorithms like k-means, and more.
- **Keras and TensorFlow** [25, 26]: Keras is a user-friendly API for building neural networks, developed in Python, and can operate on the TensorFlow framework. TensorFlow is a machine learning framework developed by Google. These libraries are used for building and training deep learning models for their ability to handle complex patterns in large datasets.
- **PM4Py** [27]: PM4Py is a process mining library in Python. It provides tools for the automatic discovery of process models from event logs.
- **PyArrow** [28]: PyArrow is a cross-language development platform for in-memory data. It provides a standardized columnar memory format that can be used to accelerate the processing of large datasets.
- **Hmmlearn** [29]: Hmmlearn is a library for Hidden Markov Models (HMMs) in Python.

- **Matplotlib** [30]: Matplotlib is a library for creating static, animated, and interactive visualizations in Python. It is widely used for visualizing datasets, identifying patterns, and presenting analysis results through graphs and diagrams.
- **Datetime** [31]: The Datetime library is used for manipulating dates and times. It helps in converting data into various date and time formats.

Chapter 3

Approach

This chapter details the process of developing an algorithm for NILM. Initially, a regression approach was considered to solve the NILM problem. However, after developing the initial algorithm, the results were irregular and produced unexpected patterns for some devices. Consequently, the models were restructured to address the problem as a classification task.

The chapter begins with an examination of the dataset, including preprocessing steps, initial analysis, and feature selection. It then describes the process of preparing the data for disaggregation. Finally, the application of machine learning and other mathematical methods for NILM is discussed. The methods used in this research are available on GitHub at the following link: GitHub Repository.

3.1 Dataset Research

The foundation of a NILM system is an accurately labeled dataset, including both the aggregated energy consumption and the individual contributions of each device. Initially, there was an idea to develop our own dataset by measuring various appliances under different configurations. However, this approach was deemed resource-intensive, leading to a preference for leveraging existing datasets. Below are the datasets found during the research, all of which measure only active power:

- **UK-DALE**: The UK-DALE dataset includes data from five distinct households in England. It is sampled once every 6 seconds, with additional sam-

pling at 16 kHz for three of the houses. Each house spans an extended period and offers detailed recordings, providing diversity due to different household configurations, occupancy patterns, and lifestyle choices [32].

- **REDD**: The Reference Energy Disaggregation Data Set (REDD) is a well-known dataset in the NILM community, originating from the United States. It features power consumption data from six households, collected at both high and low frequencies. High-frequency data are sampled at 15 kHz, while low-frequency data are recorded every second. REDD was used as a foundation for many initial disaggregation algorithms [33].

- **AMPds**: The Almanac of Minutely Power dataset (AMPds) contains detailed energy, water, and natural gas measurements from a single household in Canada over two years. It includes data from 21 different circuits, sampled at one-minute intervals. This dataset is rich for granular analysis [34].

- **iAWE**: The India Adaptive Workload Experiment (iAWE) dataset offers energy consumption data from an Indian household, capturing not only electrical parameters but also environmental conditions such as temperature and humidity. It records data at one-second intervals. This dataset is particularly interesting for NILM studies due to its inclusion of power outages and micro-generation events, which are common in many developing regions [35].

- **GREEND**: The GREEND dataset provides energy consumption data from households across Austria and Italy, sampled at one-second intervals. It highlights the variability in energy usage patterns across different climates and cultural contexts within Europe. The dataset focuses on the power consumption of individual appliances and lighting loads in residential settings [36].

Furthermore, the emergence of synthetic datasets opens new ways for NILM exploration by offering simulated energy consumption scenarios. These datasets can be important for testing in more controlled environment where ground truths are precisely known, which can be challenging to achieve with real-world data. While real-world datasets capture the complexity and unpredictability of actual

energy consumption patterns, synthetic datasets facilitate the validation and testing of NILM algorithms under idealized conditions. This enables researchers to systematically evaluate algorithm performance and robustness before applying them to more complex, real-world scenarios. One example of such dataset is SynD [37].

An exploration of available datasets favored those originating from Europe, as this thesis is written in Norway and involves similar electrical devices and power frequency standards. Among the researched datasets, two were collected in Europe: UK-DALE and GREEND. Both of them are very detailed, but UK-DALE was chosen as the most suitable. It contains a greater variety of household devices, and its data format is divided into individual files for each device. This structure provides additional flexibility, allowing for the merging and processing of the data in a manner that best suits the needs.

3.2 Preprocessing and Data Analysis

The UK-DALE dataset contains data from five houses, each with different configurations, patterns, and lifestyles. The first house contains 52 different appliances, providing extensive control over all devices and electricity usage. Houses two and five each contain around 20 different devices, while houses three and four contain only 4 or 5 devices each.

3.2.1 Preprocessing

After selecting a folder corresponding to a house and analyzing the first 10 lines of data for some devices and the aggregated power, it was observed that the timestamps are not synchronized across all devices. The aggregated power is sampled at a higher frequency, while the devices are sampled at varied intervals of around 10 seconds. Therefore, resampling is necessary. The devices, along with the aggregated load, were then loaded into a single DataFrame using Pandas by merging on the timestamp and resampling to one-minute intervals.

For this reason, both mean() and first() resampling methods were used. The mean() method calculates the mean of all samples within a given interval, while the first() method takes the first value and uses it for the entire interval. After some testing, it was noticed that first() would be more accurate as it does not

generate non-existent loads, which can introduce inaccuracies. For example, an oven that operates on two energy modes, low and high, could create a medium mode if the modes change within the same interval when using mean(), potentially influencing the results. However, the first() method is also not perfectly accurate as it ignores other values within the interval.

After merging and filling null and corrupted values, Figure 3.1 shows a data frame with a timestamp, aggregated power of all devices, and power of some individual devices, measured in Watts.

	timestamp	aggregate	stereo_speakers_bedroom	i7_desktop	hairdryer	primary_tv	2
0	2014-06-29 16:23:00	768.0	1.0	114.0	0.0	69.0	
1	2014-06-29 16:24:00	777.0	1.0	118.0	0.0	69.0	
2	2014-06-29 16:25:00	2285.0	1.0	113.0	1.0	68.0	
3	2014-06-29 16:26:00	769.0	1.0	115.0	0.0	68.0	
4	2014-06-29 16:27:00	760.0	1.0	115.0	0.0	69.0	

Figure 3.1: Snippet of DataFrame of UK-DALE house five.

3.2.2 Analysis of Devices

As explained in the Background chapter 2, electrical devices have different usage patterns; some devices are turned on once per month on an irregular basis, and some are harder to recognize than others. Therefore, it is necessary to analyze the dataset to determine which devices should be used for further analysis and which should be dismissed. The criteria for selecting devices are that they should be used on a regular basis and should not be variable-state devices. For this analysis, we use Matplotlib to visualize when a given device is on/off and how much electricity (in Watts) it uses.

Figure 3.2 shows a scatter plot of hairdryer usage over a month. Each dot on the graph corresponds to one minute, with the timeline containing 43,200 intervals, which is one month (60 minutes * 24 * 30). From the graph, we can see two operating states: off and on. The ”on” state is around 1000 Watts, typically last-

ing for 1-3 minutes. There are two dots at 400 and 70 Watts, which could indicate a low operating mode of device or a measurement error. For clear analysis, these two are not considered as an extra mode. Thus, it is a two-state device, which is advantageous for aggregation problems.

The other criterion is if there are visible regular cycles as they are easier to identify in NILM systems because their power consumption patterns are repetitive and predictable. By analyzing these cyclical patterns, NILM algorithms can more accurately disaggregate and identify the contributions of such devices. On the Figure 3.2 can be seen that cycles are existing.

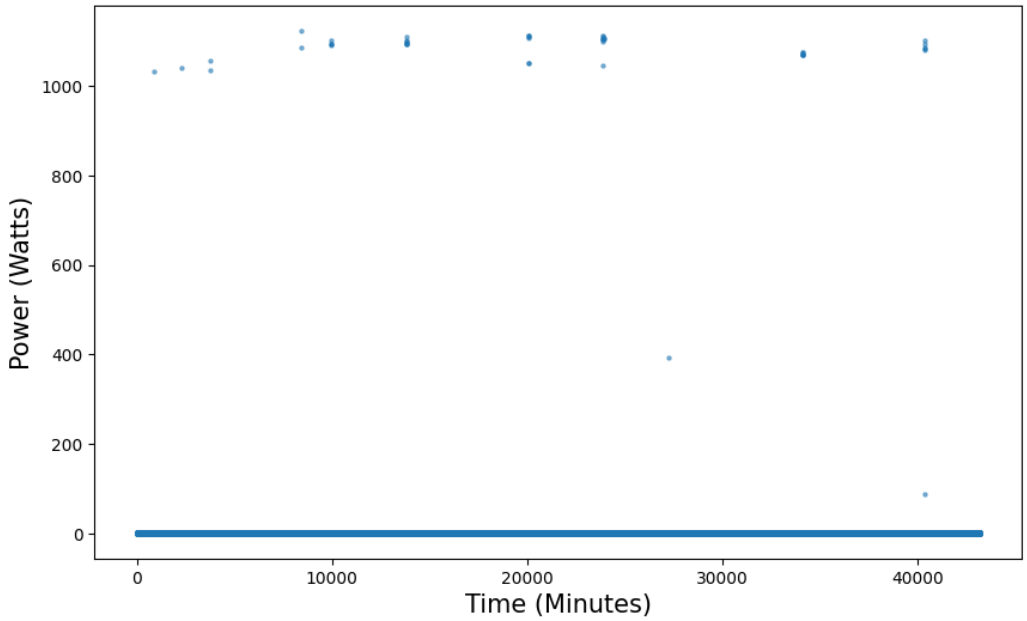


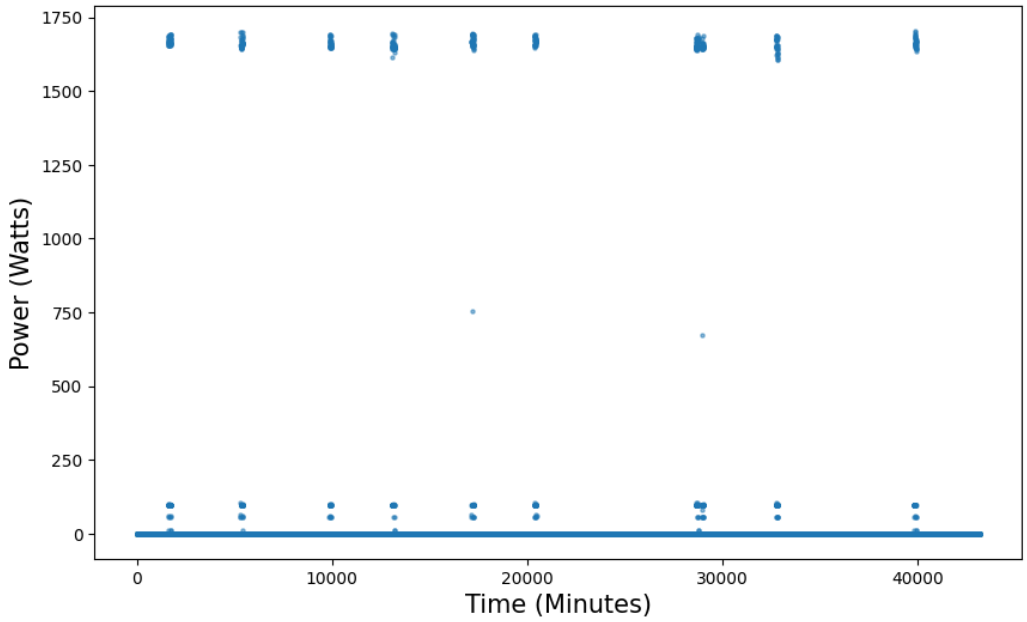
Figure 3.2: Usage of hairdryer over a month.

Figure3.3a presents an analysis of a multi-state device, specifically a dishwasher. It depicts the power consumption of the dishwasher over the course of a month. The scatter plot reveals three distinct operating states. The high power state, around 1600 Watts, likely corresponds to the main washing and heating phases of the dishwasher cycle. Additionally, there are two lower power states visible around 100 Watts and 70 Watts, which correspond to the water filling/rinsing and drying phases, respectively.

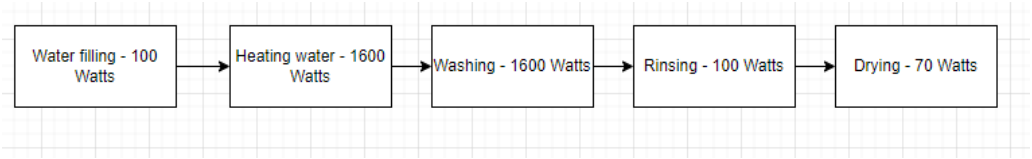
Figure 3.3b shows an illustrative diagram outlines the typical sequence of cy-

cles in a dishwasher. This diagram was informed and corrected based on the empirical data observed in the scatter plot of dishwasher. The cycle starts with the Water Filling phase, consuming around 100 Watts. This is followed by the Heating Water phase, which uses a substantial amount of power (approximately 1600 Watts) to prepare for the main cleaning cycle. The Washing phase, maintaining the high power usage, ensures thorough cleaning of the dishes. Subsequently, the Rinsing phase, returning to around 100 Watts, removes any remaining detergent. Finally, the Drying phase, operating at a lower power level of about 70 Watts, completes the cycle.

The repetitive nature of these cycles with fixed amount of operating states makes the dishwasher an excellent candidate for NILM research.



(a) Usage of dishwasher over a month.



(b) Order of cycles in a typical dishwasher.

Figure 3.3: (Top) Usage of dishwasher over a month. (Bottom) Order of cycles in a typical dishwasher.

The final example of a device is a computer. The scatter plot of its power usage is shown in Figure 3.4. From the plot, it can be observed that the computer is running continuously. Unlike devices with distinct operating cycles, the computer's power consumption fluctuates significantly throughout the day, indicating it is a variable-state device. This variability can be caused by the computer performing different tasks that require varying amounts of power, such as running applications, processing data, or using external devices like speakers or hard drives.

This inherent variability and lack of visible repetitive patterns make computers challenging to classify accurately in NILM studies.

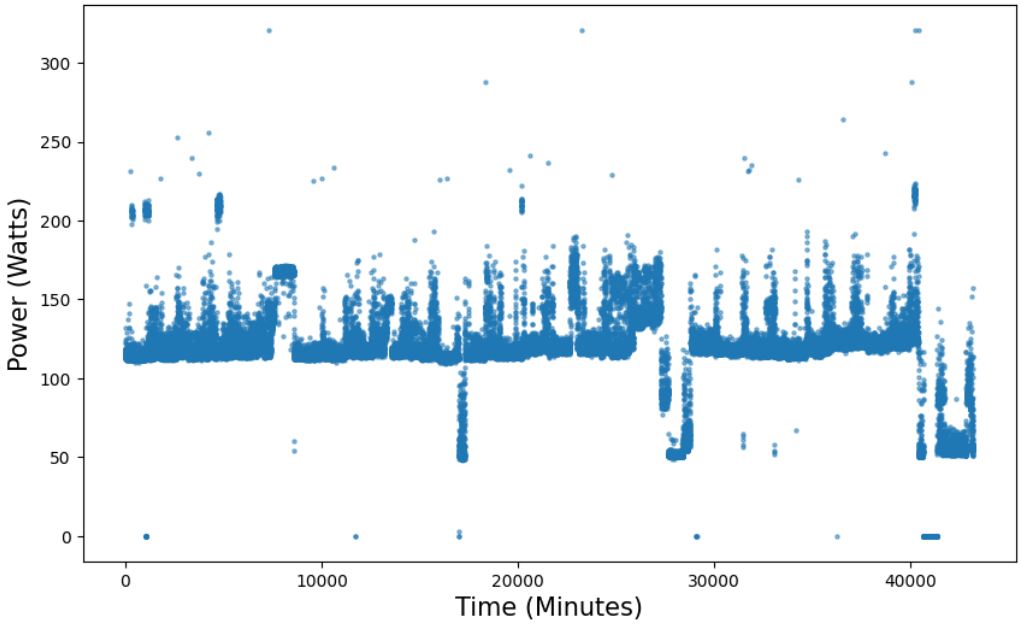


Figure 3.4: Usage of computer over a month.

In the ways shown in these subsection examples, all devices are classified to determine their suitability for NILM research, and those that do not meet the criteria are excluded from further analysis. Additionally, their impact on the aggregated power load is disregarded.

3.2.3 Data Clustering

The next step is to cluster the data using k-means clustering. Instead of predicting the exact amount of electricity in Watts a given device is using at any given

point, we simplify the analysis by predicting the operating state of the device. This approach reduces continuous power measurements to discrete states, making it easier to identify and analyze patterns in the data. Given that devices have a fixed number of operating states, the k-means algorithm effectively clusters their power consumption data into these distinct states.

For example, in Figure 3.2, the hairdryer has only two operating states. After applying the clustering algorithm, the dataset replaces the continuous power values with discrete states: 0 for ”off” and 1 for ”on”, where the ”on” state corresponds to around 1000 Watts.

Figure 3.5 shows the clustered usage of an oven. Colors indicate different operating states. For the ”off” state, the oven is classifies also standby mode, using a threshold of 20 Watts. The plot reveals a medium mode around 2200 Watts and a high mode around 3000 Watts, indicating that the user is adjusting the oven’s heating settings. There is also a third state around 100 Watts, which represents the oven holding its temperature once the desired mode is reached. Therefore, the clustered DataFrame for the oven would have numbers 1 to 3 for the ”on” modes and 0 for the ”off/standby” mode.

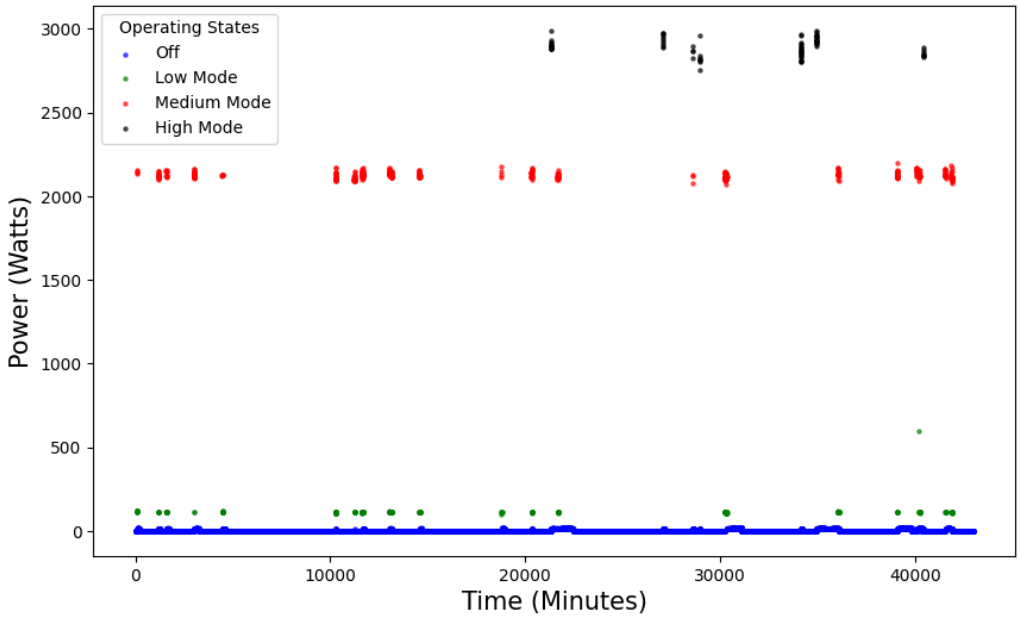


Figure 3.5: Clustered usage of oven over a month.

Once all devices are clustered according to their usage patterns, the preprocessing step is complete. The result is a DataFrame that includes the chosen devices and their clustered states of usage, as shown in Figure 3.6.

	timestamp	aggregate	hairdryer	primary_tv	fridge_freezer	washer_dryer
0	2014-06-29 16:23:00	769.000000	0	2	3	1
1	2014-06-29 16:24:00	1077.700000	0	2	3	2
2	2014-06-29 16:25:00	2264.444444	0	2	3	3
3	2014-06-29 16:26:00	763.500000	0	2	3	1
4	2014-06-29 16:27:00	1980.400000	0	2	3	3

Figure 3.6: Snippet of DataFrame with clustered devices.

3.3 Feature Selection

Effective feature selection is crucial for the performance of machine learning models. In the context of NILM, the initial data frame includes two columns that serve as the primary inputs for machine learning models: timestamp and aggregated power load. However, these columns alone may not be sufficient, as timestamps are unique and the aggregated power load can vary slightly each time the same device is on. Machine learning models are designed to handle irregular input values, but their efficiency can be significantly improved by adding more structured and repetitive features.

To enhance the predictive power of the models, additional features can be extracted from the timestamp column:

- **Hour of the Day**: extracting the hour from the timestamp provides a feature that captures daily usage patterns. This is done using the datetime package.
- **Day of the Week**: This feature captures weekly usage patterns by assigning numbers 0 to 6 for Monday to Sunday. This can also be extracted using the datetime package.
- **Weekend Indicator**: This binary feature indicates whether a given day is a weekend or a weekday. For weekdays (Monday to Friday), the value is 0, and for weekends (Saturday and Sunday), the value is 1.

These additional features introduce repetitiveness into the dataset, making it easier for machine learning algorithms to identify and analyze patterns.

Furthermore, to improve the efficiency of the models, clustering techniques can be applied to the aggregated power load column. By clustering the power load into discrete levels (e.g., low, medium, high), the models can operate on more regular and structured input data. This process involves:

- **Clustering Aggregated Power Load**: Using a clustering algorithm the continuous power load values are transformed into discrete states. An extra column for the clustered load is added to the DataFrame, while retaining the original power load values for reference. For this analysis, it is assumed that the number of clusters is 5.

These feature engineering steps result in a DataFrame with multiple columns that capture temporal patterns and structured power load data, significantly enhancing the machine learning model's ability to predict device usage accurately.

Figure 3.7 illustrates the final set of feature columns used in machine learning models.

	timestamp	what_hour	what_day	is_weekend	aggregate	agg_clustered
0	2014-06-29 16:23:00	16	6	1	769.000000	2
1	2014-06-29 16:24:00	16	6	1	1077.700000	3
2	2014-06-29 16:25:00	16	6	1	2264.444444	5
3	2014-06-29 16:26:00	16	6	1	763.500000	1
4	2014-06-29 16:27:00	16	6	1	1980.400000	4

Figure 3.7: Snippet of DataFrame with selected features.

3.4 Disaggregation Methods

This section explains the methods used for load disaggregation based on the input features selected above. The objective is to predict if a device is on or off and, if applicable, its operating state. Since this is a classification problem, we use methods that support classification. Figure 3.8 shows load disaggregating concept.

We begin with a simple Random Forest model as it is straightforward to implement, copies well with overfitting and work with large datasets. It also provides

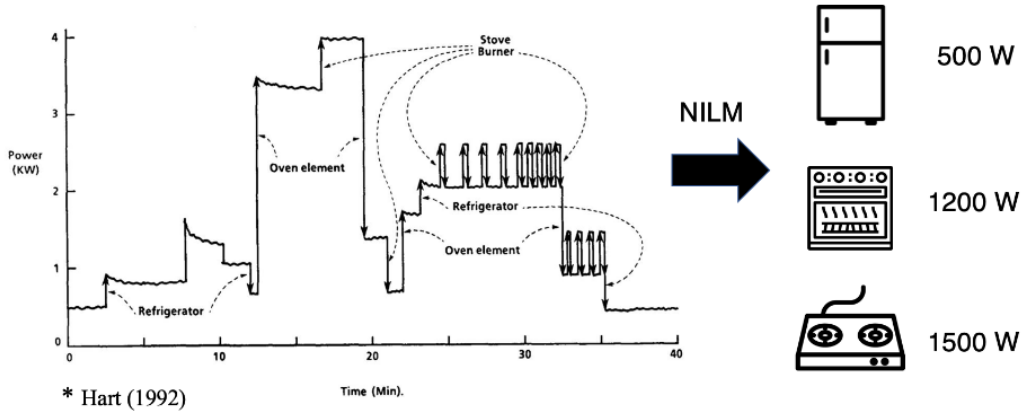


Figure 3.8: Concept of load disaggregation [38].

insights into feature importance, that helps to understand power consumption patterns.

Next, we move to more advanced machine learning models, focusing on Neural Networks (NN). Within NNs, we primarily use RNN and LSTM networks, as well as CNN.

- **RNNs and LSTMs** capture temporal dependencies in sequential data, which is beneficial for NILM due to the daily human cycles reflected in power usage. LSTMs are particularly capable of learning long-term dependencies [12].
- **CNNs** automatically extract relevant features from raw data and can be applied to time-series data to capture local temporal dependencies. They are effective in analyzing patterns in power consumption data [12].

Lastly, we incorporate process mining techniques to predict the next device and help recognize device usage patterns. This gives a valuable insights into sequential patterns of device usage.

3.4.1 Splitting the DataFrame

Before applying any disaggregation methods, the dataset is split into training and testing sets, with 70% of the data allocated for training and 30% for testing. This

standard division ensures that the models are trained on a substantial portion of the data while reserving a separate subset for evaluating the models' performance.

3.4.2 Random Forest

Random Forest is an ensemble learning method available in the `sklearn` package. Given that our problem is a classification one, we utilize `RandomForestClassifier` as the base model.

The standard configuration for `RandomForestClassifier` is to use 100 estimators, meaning that the forest consists of 100 decision trees. This configuration strikes a balance between performance and computational efficiency. Having more trees generally leads to better performance and robustness, as the ensemble of trees reduces the variance of the model and mitigates the risk of overfitting. However, increasing the number of trees also increases computational cost and memory usage, so 100 estimators provide a good compromise [39].

Since NILM is a multi-output problem, requiring the prediction of the operating states of multiple devices simultaneously, we use the `MultiOutputClassifier` wrapper. This wrapper extends the `RandomForestClassifier` to support multi-output classification by fitting one classifier per target.

Once the model is designed, the training process begins. The training data is fed into the model, and it learns to map the input features to the target labels (device operating states). During training, the Random Forest model constructs multiple decision trees, each trained on a random subset of the data. The results of these trees are aggregated to produce the final prediction.

To improve the performance of the Random Forest model, it is employed Grid Search, a hyperparameter tuning technique. Grid Search helps to systematically explore a specified parameter grid to find the combination of hyperparameters that results in the highest possible accuracy score for the `RandomForestClassifier`. The key hyperparameters we tune include the number of estimators, the maximum depth of the trees, and the minimum number of samples required to split an internal node and to form a leaf node. The Grid Search is performed using the `GridSearchCV` class from the `sklearn.model_selection` module.

In evaluation of Random Forest it is used feature importance. It is a valuable byproduct of training. It measures how valuable each feature is in constructing the decision trees. In context of NILM, feature importance will tell, which column

from selected features is the most influential in predicting the operating state of devices [39].

3.4.3 RNN/LSTM

In this approach, we aim to capture dependencies between devices by training an RNN with LSTM layers on a combined dataset rather than training separate models for each device. This novel method leverages the relationships between devices, which can be beneficial for improving the model's performance. While previous studies, such as those referenced in Article 2.9.2, employ deep learning algorithms like LSTM to train models for each device independently, they fail to account for inter-device relationships. By integrating data from all devices into a single data frame, our model can identify patterns and dependencies across devices.

The dataset is structured to represent the states of multiple devices. Each device, having a specific number of operating states, is encoded using one-hot encoding. One-hot encoding transforms categorical variables into binary vectors, where each state is represented by a vector with a single "hot" (1) element and the rest "cold" (0). For example, Figure 3.9 illustrates the one-hot encoding process

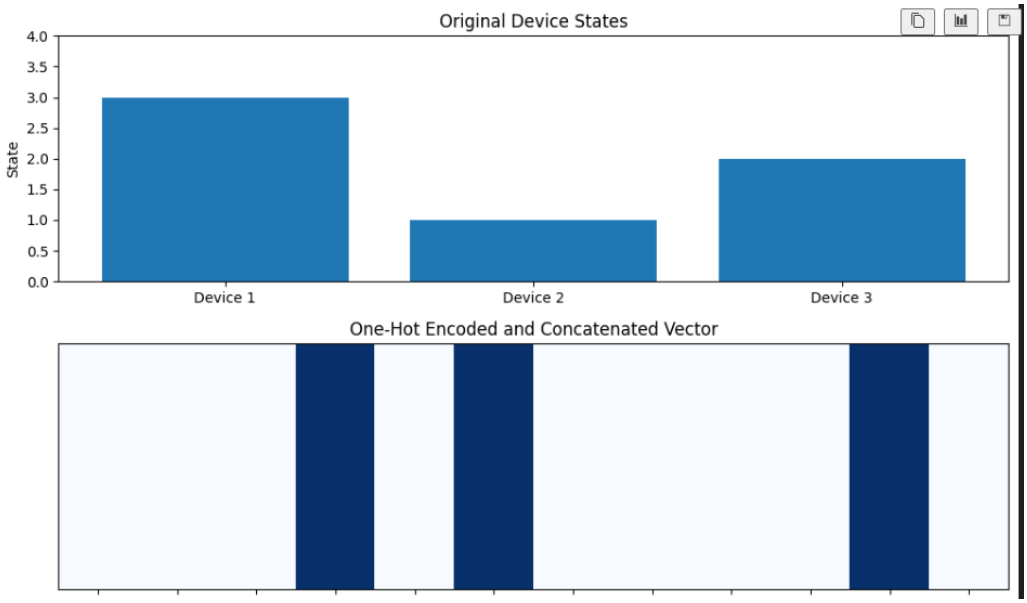


Figure 3.9: Process of one hot encoding.

for three devices with four states each, converted into binary vectors [40].

Data Splitting

To ensure robust model training and evaluation, the dataset is divided into training, validation, and test sets in the following proportions: 70% for training, 15% for validation, and 15% for testing. This split allows for model tuning on the validation set and unbiased performance assessment on the test set.

Model Architecture

The proposed model employs two LSTM layers to capture temporal dependencies and inter-device relationships. The model was designed by experimenting with different configurations, ultimately selecting the one that provided the best results [41]. The architecture is made with 'TensorFlow' package as follows:

- **Input Layer**: Shape (1440, 5), where 1440 represents the sequence length (e.g., minutes in a day if data is sampled per minute) and 5 represents the features or columns in the input data.
- **First LSTM Layer**: This layer consists of 64 units and returns sequences, enabling the next LSTM layer to receive sequence data.
- **Dropout Layer**: A dropout rate of 0.3 is used to prevent overfitting by randomly setting 30% of the input units to 0 during training.
- **Second LSTM Layer**: Similar to the first, this layer also consists of 64 units and returns sequences to maintain temporal data flow.
- **Output Layer**: A Dense layer with 9 units (representing 3 devices, each with 3 states) and a softmax activation function. The softmax function outputs a probability distribution over the states of each device.

The complete architecture is depicted in Figure 3.10.

Model Compilation and Training

The model is compiled using the 'categorical crossentropy' loss function. Categorical crossentropy is suitable for multi-class classification problems where each example belongs to one of several classes. This choice is appropriate as it measures the performance of the model whose output is a probability distribution over multiple classes.

The optimizer used is 'adam', a popular optimization algorithm that adapts the learning rate during training, resulting in faster convergence. The model is

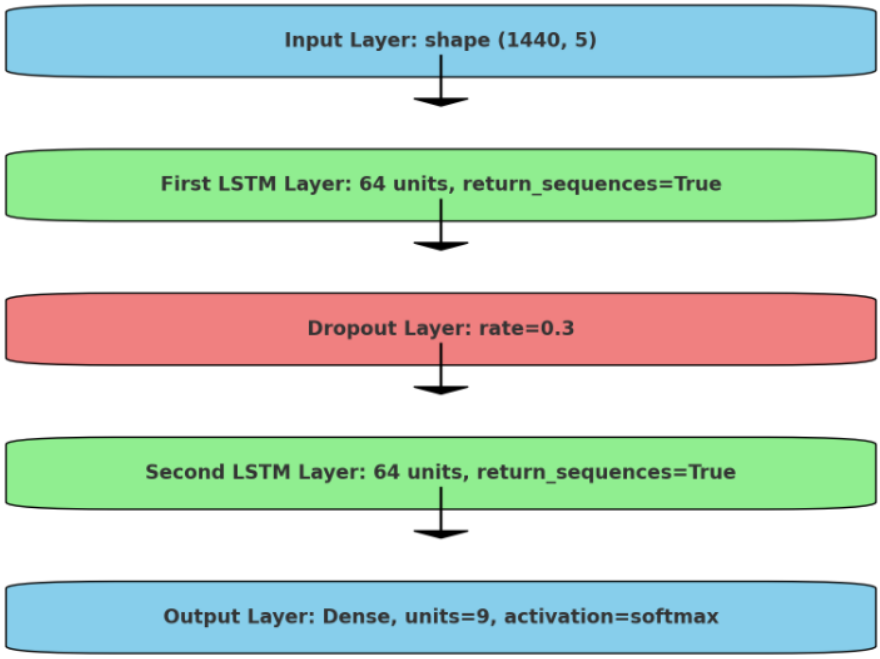


Figure 3.10: LSTM Neural Network Model Architecture.

trained for 30 epochs, which is determined to be optimal for balancing training time and performance improvement [41].

3.4.4 CNN

Designing the CNN, we use the same input data as for the RNN with LSTM layers. The model was designed by experimenting with different configurations [42].

The architecture of this CNN is designed using 'TensorFlow' package as follows:

- **Input Layer**: The input layer has a shape of (1440, 5), where 1440 represents the sequence length (e.g., minutes in a day if data is sampled per minute) and 5 represents the features or columns in the input data.
- **First Conv1D Layer**: This layer has 64 filters with a kernel size of 3 and uses the ReLU activation function. It is used a 1D convolutional layer be-

cause our input data is sequential, with each time step represented as a feature vector.

- **Dropout Layer**: A dropout rate of 0.3 is used to prevent overfitting by randomly setting 30% of the input units to 0 during training.
- **Second Conv1D Layer**: This layer has 128 filters with a kernel size of 3 and uses the ReLU activation function. Increasing the number of filters allows the model to capture more complex patterns in the data.
- **Dropout Layer**: Another dropout layer with a rate of 0.3 is used for regularization to prevent overfitting.
- **Flatten Layer**: This layer flattens the 2D matrix output from the convolutional layers into a 1D vector, making it suitable for input into the dense layers.
- **Dense Layer**: This fully connected layer has 64 units and uses the ReLU activation function. The dense layer processes the flattened input to learn more complex representations of the data.
- **Dropout Layer**: A dropout rate of 0.5 is used to prevent overfitting by randomly setting 50% of the input units to 0 during training.
- **Output Layer**: The output layer is a dense layer with 9 units (representing 3 devices with 3 states each) and uses the softmax activation function. The softmax function outputs a probability distribution over the states of each device.

The complete architecture is depicted in Figure 3.11.

The model is compiled similar to the previous model using the 'categorical crossentropy' loss function and with 'adam' optimizer. The model is trained for 30 epochs.

3.4.5 Process Mining

In this part, we aim to find correlations between devices, such as whether there are sequences of device operations or if some devices consistently operate simultaneously. This analysis is performed using the 'pm4py' package, which is a Python library for process mining.

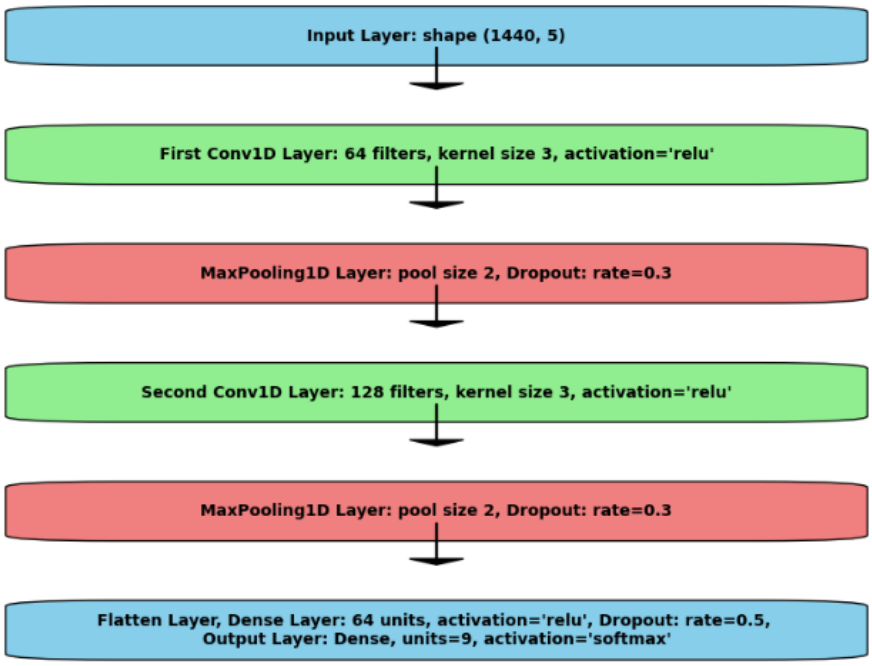


Figure 3.11: CNN Model Architecture.

First, we convert the data frame into event logs, including attributes such as timestamp, device name, and state. Each row in the event log corresponds to a different device, recording its energy usage over time. This conversion is depicted in Figure 3.12.

	case:concept:name	concept:name	time:timestamp
0	i7_desktop	i7_desktop_high	2014-06-30 16:23:00+00:00
1	hairdryer	hairdryer_off	2014-06-30 16:23:00+00:00
2	primary_tv	primary_tv_off	2014-06-30 16:23:00+00:00
3	24_inch_lcd_bedroom	24_inch_lcd_bedroom_off	2014-06-30 16:23:00+00:00
4	treadmill	treadmill_off	2014-06-30 16:23:00+00:00
5	fridge_freezer	fridge_freezer_very_high	2014-06-30 16:23:00+00:00

Figure 3.12: Event logs.

Once the event logs are prepared, we apply the Alpha Miner algorithm, a well-

known process discovery method in process mining. The Alpha Miner algorithm helps to identify the process model by discovering places, transitions, and arcs, which are crucial for understanding the workflow and interactions between different devices.

- **Places**: Represent the states or conditions in the process model where something may happen.
- **Transitions**: Represent the events or activities that cause a change in state.
- **Arcs**: Represent the flow between places and transitions, indicating the order and dependency of events.

The application of the Alpha Miner algorithm allows us to visualize and analyze the process model, identifying patterns such as concurrent device operations or sequential dependencies [17].

3.5 Evaluation Methods

For evaluating machine learning models, we use below metrics:

Accuracy is used to evaluate the overall correctness of the model by checking the predicted values of all devices across the given timestamps. This metric is particularly useful for understanding the model's general performance.

Hamming accuracy, on the other hand, checks each value individually to determine if it is correct. It provides a more granular view of the model's performance by evaluating the accuracy at each specific point rather than the overall sequence.

Additionally, we use 'matplotlib' to visualize the predicted and original values, allowing for a direct comparison between them.

Chapter 4

Experimental Evaluation

4.1 Experimental Setup and Data Set

The dataset utilized for this experiment is the UK Domestic Appliance-Level Electricity (UK-DALE) dataset. In this study, data from two houses (House 5 and House 2) were selected for analysis:

- **House 5**: Six specific devices were chosen and then clustered based on criteria outlined in Sections Sections 3.2.2 and 3.2.3. The selected devices are:
  - Hairdryer
  - Bedroom TV
  - Kettle
  - Oven
  - Dishwasher
  - Washing Machine
- **House 2**: Also six devices were chosen and clustered:
  - Kettle
  - Rice Cooker
  - Washing Machine
  - Dishwasher

  - Microwave
  - Toaster

Both datasets are approximately 200,000 rows in size, which corresponds to about half a year of one-minute intervals.

Once the devices were selected, the following steps were taken, as detailed in the approach section 3:

- The power consumption values of the selected devices were summed to create an aggregate power usage profile for each house.
- Various features were extracted from the aggregated data to capture temporal and consumption patterns.
- These features were then used as inputs for training the machine learning models.

4.2 Experimental Results

In this section, the results from all aggregation methods presented in Section 3.4 are shown.

4.2.1 Random Forest

Using GridSearch, we determined that the best parameters for both models (two houses) are: `RandomForestClassifier(n_estimators=50, min_samples_leaf=1, min_samples_split=2, max_depth=None, random_state=42)`.

House 5

For house 5, feature importance analysis revealed that the column 'aggregate' has the highest influence on the output, followed by 'what_hour' and 'agg_clustered'. The least influence is from whether it is a weekend or not, at 0.6%. Detailed average percentages from all devices are shown below:

- **aggregate**: 0.5221
- **what_hour**: 0.1951

- **agg_clustered**: 0.2422
- **what_day**: 0.0483
- **is_weekend**: 0.0063

The general accuracy, along with detailed accuracy for each device, is shown below:

```
Exact Match Accuracy: 98.67%
Hamming Accuracy for Label 1 (24_inch_lcd_bedroom): 99.38%
Hamming Accuracy for Label 2 (dishwasher): 99.63%
Hamming Accuracy for Label 3 (hairdryer): 99.99%
Hamming Accuracy for Label 4 (kettle): 99.95%
Hamming Accuracy for Label 5 (oven): 99.72%
Hamming Accuracy for Label 6 (washer_dryer): 99.57%
Overall Hamming Accuracy: 99.71%
```

For a more detailed analysis, one device was chosen and analyzed. Below is the detailed analysis of the 'washer_dryer', showing details about each state of the device such as precision, recall, f1-score, and support. This information provides insights into how well each state is recognized by the model.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56557
1	0.95	0.88	0.91	1241
2	0.97	0.99	0.98	751
3	0.99	0.99	0.99	581
4	0.87	0.69	0.77	84
accuracy			1.00	59214
macro avg	0.96	0.91	0.93	59214
weighted avg	1.00	1.00	1.00	59214

The lowest precision is observed in state 4, with 87% precision occurring 84 times across the testing dataset. The highest precision is in the off (0) state, with 100% precision occurring 56,557 times.

Figure 4.1 shows a scatter plot of the 'washer_dryer' device clustered into 5 states. Each dot represents the original value before prediction. Green dots indicate correct predictions, while red dots indicate mismatches.

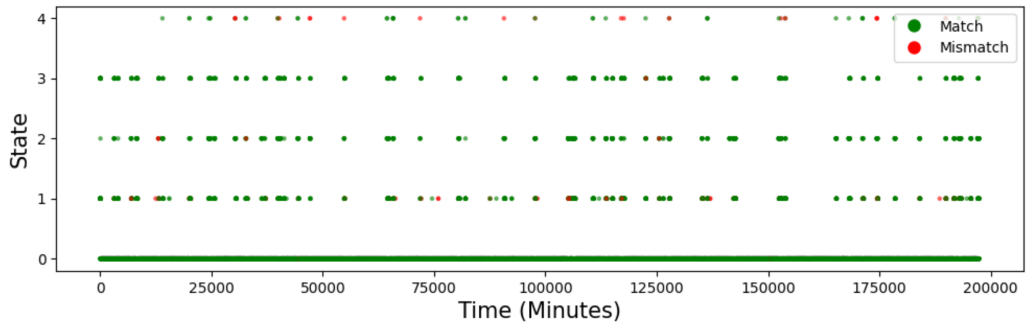


Figure 4.1: Scatter plot of washer_dryer states predicted with Random Forest.

House 2

For house 2, feature importance analysis is similar to that for House 5. The highest influence is again the 'aggregate' column with more than 50%. The second is 'agg_clustered' with 32%, and third is 'what_hour' with 11%. The least influence is whether it is a weekend or not, at 0.07%. Detailed average percentages from all devices are shown below:

- **aggregate**: 0.5387
- **what_hour**: 0.1183
- **agg_clustered**: 0.3210
- **what_day**: 0.0142
- **is_weekend**: 0.0072

The general accuracy, along with detailed accuracy for each device, is shown below:

```
Exact Match Accuracy: 99.75%
Hamming Accuracy for Label 1 (dishwasher): 99.89%
Hamming Accuracy for Label 2 (kettle): 99.99%
Hamming Accuracy for Label 3 (microwave): 99.90%
Hamming Accuracy for Label 4 (rice_cooker): 99.98%
Hamming Accuracy for Label 5 (toaster): 99.99%
Hamming Accuracy for Label 6 (washing_machine): 99.88%
Overall Hamming Accuracy: 99.94%
```

Again, for a more detailed analysis, one device was chosen and analyzed. Below is the detailed analysis of the ‘dishwasher‘.

```
Classification Report for dishwasher:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     57904
           1       0.97      0.96      0.97       419
           2       0.99      0.98      0.98      1007
           3       0.00      0.00      0.00         4

    accuracy                           1.00     59334
   macro avg       0.74      0.73      0.74     59334
weighted avg       1.00      1.00      1.00     59334
```

The dishwasher with 4 operating states gives 99.89% Hamming accuracy. The first three states are almost always correctly identified, while the fourth state is not well identified, occurring only 4 times in the test set.

Graph 4.2 shows a scatter plot of the ‘dishwasher‘ device. Each dot represents the original value before prediction. Green dots indicate correct predictions, while red dots indicate mismatches.

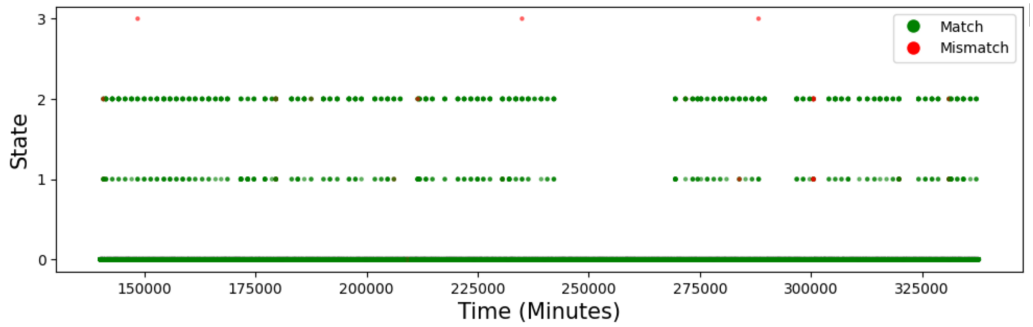


Figure 4.2: Scatter plot of dishwasher states predicted with Random Forest.

Summary

Figure 4.3 shows the averaged results from both houses with the chosen two devices.

	HOUSE 5	HOUSE 2	AVERAGE
Accuracy	98.67%	99.75%	99.21%
Overall Hamming	99.71%	99.92%	99.815%
Dishwasher	99.63%	99.89%	99.76%
Wash Machine	99.57%	99.88%	99.725%

Figure 4.3: Averaged results from both houses using Random Forest.

4.2.2 RNN/LSTM

For RNNs using LSTM layers, various configurations were tested. Initially, the model was trained on sequences with a window size of 1440 (one day). However, this setup only returned 'off' states for all devices, indicating that the model couldn't detect any meaningful patterns and was defaulting to the 'off' state.

After adjusting the window size, the best results were obtained with a window size of 60 (one hour) and a step size of 30. Using a step size of 30 effectively doubled the amount of data, as each window overlaps by half with the next. Although the general accuracy difference between step sizes of 30 and 60 was 0.5%, using a step size of 30 doubled the size of the training and testing data frames. Despite a 15% validation split, the size of the testing data remained the same as in the random forest model, which did not use validation.

Data from two houses were used for training.

House 5

The general accuracy and per-device accuracy are shown below for two configurations: first, an LSTM with 64 neurons, and second, with 128 neurons. The model with 128 neurons produced better results overall and for each appliance.

64 LSTM

```
Exact Match Accuracy: 91.30%
Hamming Accuracy for Label 1 (hairdryer): 99.88%
Hamming Accuracy for Label 2 (24_inch_lcd_bedroom): 97.74%
Hamming Accuracy for Label 3 (kettle): 99.81%
Hamming Accuracy for Label 4 (oven): 98.22%
Hamming Accuracy for Label 5 (dishwasher): 98.09%
Hamming Accuracy for Label 6 (washer_dryer): 96.34%
Overall Hamming Accuracy: 98.35%
```

128 LSTM

```
Exact Match Accuracy: 95.91%
Hamming Accuracy for Label 1 (hairdryer): 99.96%
Hamming Accuracy for Label 2 (24_inch_lcd_bedroom): 97.63%
Hamming Accuracy for Label 3 (kettle): 99.90%
Hamming Accuracy for Label 4 (oven): 99.43%
Hamming Accuracy for Label 5 (dishwasher): 99.11%
Hamming Accuracy for Label 6 (washer_dryer): 99.10%
Overall Hamming Accuracy: 99.19%
```

From all devices, we choose one to look closer at for comparison: 'washer_dryer', same as in the random forest model.

```
Classification Report for washer_dryer:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00     56218
           1       0.93      0.81      0.87      1237
           2       0.98      0.93      0.95       682
           3       0.97      0.85      0.91       571
           4       0.00      0.00      0.00        92

    accuracy                           0.99     58800
   macro avg       0.77      0.72      0.74     58800
weighted avg       0.99      0.99      0.99     58800
```

Figure 4.4 shows the scatter plot of the washer_dryer states predicted using the RNN with LSTM layers.

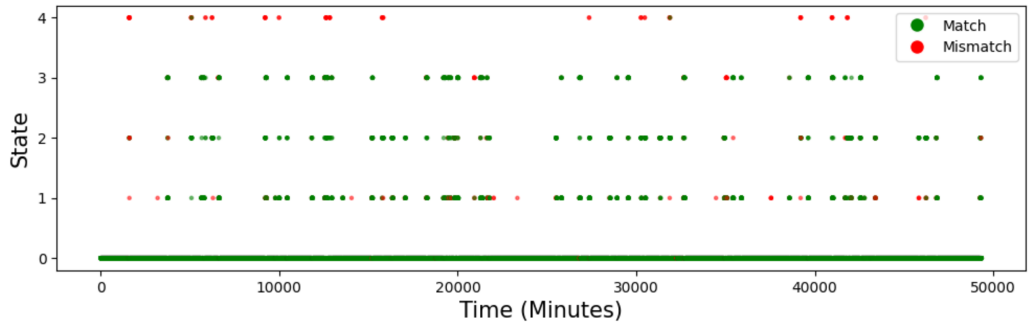


Figure 4.4: Scatter plot of washer_dryer states predicted using RNN with LSTM layers.

House 2

The general accuracy and per-device accuracy are shown below, for House 2 was used configuration with 128 neurons in LSTM layers.

```
Exact Match Accuracy: 99.38%
Hamming Accuracy for Label 1 (kettle): 99.96%
Hamming Accuracy for Label 2 (rice_cooker): 99.95%
Hamming Accuracy for Label 3 (washing_machine): 99.72%
Hamming Accuracy for Label 4 (dish_washer): 99.85%
Hamming Accuracy for Label 5 (microwave): 99.77%
Hamming Accuracy for Label 6 (toaster): 99.97%
Overall Hamming Accuracy: 99.87%
```

For detailed analysis for House 2 is chosen 'dishwasher', same as in Random Forest.

```
Classification Report for dish_washer:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     57770
           1       0.97      0.93      0.95       414
```

2	0.98	0.98	0.98	1035
3	0.00	0.00	0.00	1
accuracy			1.00	59220
macro avg	0.74	0.73	0.73	59220
weighted avg	1.00	1.00	1.00	59220

Figure 4.5 show graph for this appliance.

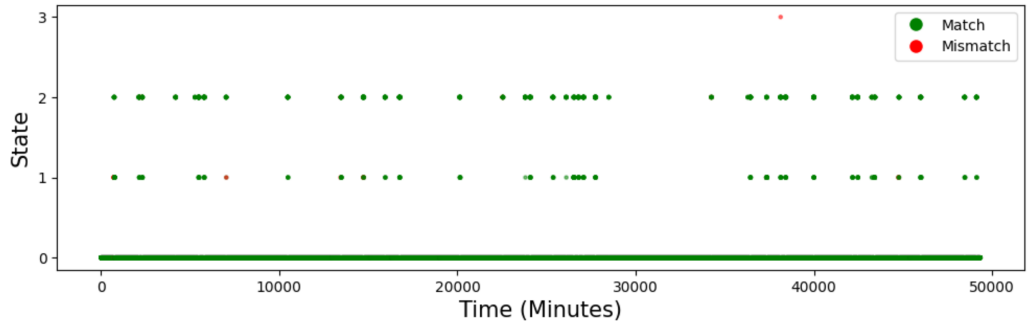


Figure 4.5: Scatter plot of dishwasher states predicted with RNN using LSTM layers.

Summary

Table visible on Figure 4.6 show averaged results from both houses and two selected devices.

4.2.3 CNN

After testing various configurations of the CNN model, including adding extra convolutional layers, dropout, and max-pooling layers, the model could not learn the patterns of the devices. It consistently returned the 'off' state for all devices. This behavior was observed for both houses. Therefore, for brevity, we present the results from House 5 only, as both houses exhibited similar outcomes.

The general accuracy, along with detailed accuracy for each device from House 5, are shown below:

```
Exact Match Accuracy: 88.45%
```

	HOUSE 5	HOUSE 2	AVERAGE
Accuracy	95.91%	99.38%	97.65%
Overall Hamming	99.19%	99.87%	99.53%
Dishwasher	99.11%	99.85%	99.48%
Wash Machine	99.10%	99.72%	99.41%

Figure 4.6: Averaged results from both houses using RNN with LSTM layers.

```
Hamming Accuracy for Label 1 (hairdryer): 99.82%
Hamming Accuracy for Label 2 (24_inch_lcd_bedroom): 96.33%
Hamming Accuracy for Label 3 (kettle): 99.82%
Hamming Accuracy for Label 4 (oven): 98.09%
Hamming Accuracy for Label 5 (dishwasher): 97.78%
Hamming Accuracy for Label 6 (washer_dryer): 95.61%
Overall Hamming Accuracy: 97.91%
```

Below, we provide detailed information for one chosen device, 'washer_dryer', and Figure 4.7 visualizes the predictions for this device.

```
Classification Report for washer_dryer:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98     56218
           1       0.00      0.00      0.00      1237
           2       0.00      0.00      0.00       682
           3       0.00      0.00      0.00       571
           4       0.00      0.00      0.00        92

    accuracy                           0.96     58800
   macro avg       0.19      0.20      0.20     58800
weighted avg       0.91      0.96      0.93     58800
```

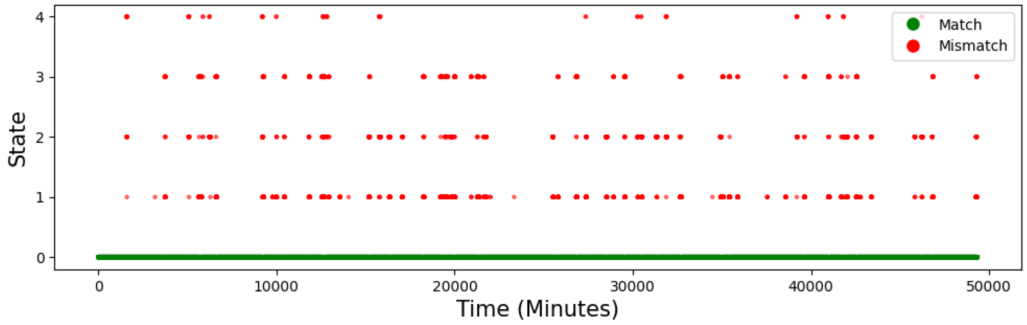


Figure 4.7: Scatter plot of washer_dryer states predicted with CNN.

4.2.4 Process Mining

Using process mining to find connections between shows reveals no relations between devices for both houses. However, the algorithm could identify connections within individual devices. The device with the most evident internal connections was the washing machine, which operates in cycles, indicating that its states change in a specific order. When the washing machine turns on, it always progresses to a low mode of electricity consumption, then to a higher mode, and finally turns off. These transitions represent the cycles of how the washing machine operates.

Figure 4.8 shows the Petri Net visualization of the process mining results. For all devices except the washing machine, their cycles start at 'start,' move to a state, and end at 'end.' In contrast, the washing machine's cycle passes through different states, reflecting its operational stages. This pattern is visible in the configurations for both house 2 and house 5.

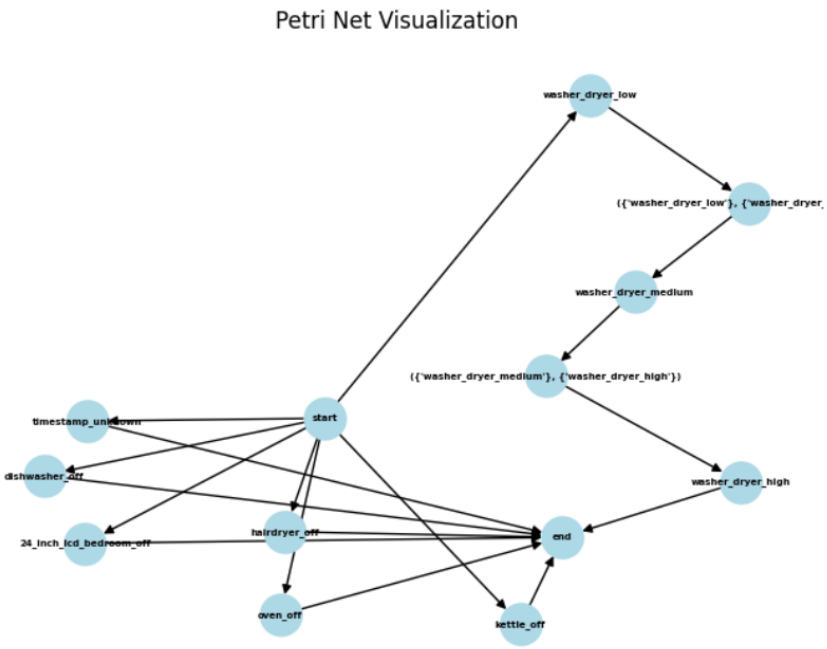


Figure 4.8: Petri Net Visualization

Chapter 5

Discussion

5.1 Objectives

The primary objective of this thesis is to find reliable methods for identifying electrical devices in a household based on aggregate electrical signals from a main electric monitor. This involves several key steps:

1. **Dataset Creation and Acquisition**: Creating or obtaining a dataset that can be used to train algorithms for device identification.
2. **NILM Procedure**: Implementing the standard Non-Intrusive Load Monitoring (NILM) procedure, which includes preprocessing data, feature extraction, event detection, and load disaggregation.
3. **Machine Learning Methods for Load Disaggregation**: Utilizing machine learning methods, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs/LSTMs), to determine their effectiveness in learning and identifying device usage patterns.

After developing a reliable NILM method, the goal is to create a working model that can be integrated into an application for optimizing smart home systems. This includes identifying anomalies in the network and providing consumers with insights into household energy usage.

5.2 Results Overview

This study utilized the UK-DALE dataset to apply various machine learning methods for disaggregation. Three machine learning methods were chosen for disaggregation: Random Forest, RNN/LSTM, and CNN, alongside process mining to better understand inter-device relationships. Experiments yielded varying results across these models.

5.2.1 Random Forest

The Random Forest method provided the most promising results, achieving a general accuracy of 99.21%. This score indicates the percentage of correctly identified devices at each given timestamp in the final dataset. Additionally, the Hamming accuracy, which evaluates the accuracy for each device separately, was 99.82%. Detailed analysis of selected device graphs and data showed that more samples per operating state led to higher accuracy in state prediction. Overall, Random Forest produced excellent results across all clustered groups of devices.

5.2.2 RNN/LSTM

The RNN/LSTM model, with adjusted window sizes, showed strong performance as well. Particularly, using a window size of 60 and a step size of 30 with 128 neurons for LSTM layers yielded optimal results. The general accuracy for this method was 97.65%, and the Hamming accuracy was 99.53%. Examining specific devices, such as those in House 5, revealed that the RNN/LSTM struggled with predicting the fourth state (state 4) with 0% precision for around 90 examples. In contrast, the Random Forest method predicted this state with nearly 90% accuracy. Other operating states showed similar prediction accuracies between the RNN/LSTM and Random Forest models.

5.2.3 CNN

Contrary to the other methods, the CNN model, despite multiple configurations and layers, consistently predicted the 'off' state for all devices. While it achieved a general accuracy of 88.45% and a Hamming accuracy of 97.91% for House 5, a examination revealed that the CNN provided zero precision for all operating

states except for the 'off' state, which it always chose. This issue was consistent across all devices.

5.2.4 Process Mining

Incorporating process mining techniques into the NILM method aimed to enhance results by identifying patterns between devices. However, the results indicated that patterns were found only within individual devices rather than between different devices. For instance, the algorithm could identify some patterns in the washing machine cycles, such as washing, rinsing, and water filling stages.

Overall, the Random Forest and RNN/LSTM models showed strong potential for accurate device state prediction, while the CNN model did not perform well in this context. Process mining added some value in understanding within-device patterns but did not significantly enhance inter-device relationship detection.

5.3 Interpretation of Results

The Non-Intrusive Load Monitoring (NILM) problem is inherently complex due to the unique patterns of appliance usage in each household. The results from the Random Forest and RNN/LSTM models successfully captured these patterns, thereby fulfilling a significant part of the research objectives. Adaptability was not checked in this study to determine if the same trained models can be applied to other datasets, as found available houses in UK-DALE do not have the same appliances.

The RNN/LSTM and Random Forest models' ability to capture temporal dependencies resulted in higher accuracy rates compared to the CNN model. This indicates that the grid search for Random Forest identified highly effective parameters, resulting in high efficiency. In the case of the RNN/LSTM model, parameter tuning was done through manual testing rather than an automated process, which still yielded moderately high results. Conversely, the CNN model failed to produce satisfactory results, indicating either suboptimal parameters, an inappropriate model architecture, or inherent limitations in applying CNNs to this type of problem. CNNs are primarily designed for image pattern recognition, which might explain their poor performance in this context.

A unique aspect of chosen approach was the use of a dataframe that included all devices simultaneously to train machine learning algorithms. This method differs from other studies, like Article 2.9.2 that often uses devices in isolation. By capturing dependencies between devices, our approach potentially provides a more holistic understanding of household energy usage patterns. This comprehensive setup allows the model to learn the interactions and overlapping usage patterns of different devices, contributing to the improved accuracy observed in the Random Forest and RNN/LSTM models.

The nature of chosen household appliances in this study, which operate in cycles and have multiple states, poses an additional challenge. Devices such as kettles/toasters/coffee machines operate for only a fraction of the day, making accurate state prediction more challenging. This discrepancy in operating times highlights the importance of detailed analysis beyond overall accuracy metrics.

In related literature, models often report accuracy rates around 98%, as seen in Article 2.9.1. However, results from this research indicate that high overall accuracy can be misleading if the model predominantly predicts the 'off' state, as was the case with the CNN model. This emphasizes the necessity of evaluating state precision to understand how well each model predicts the various operating states of devices. By examining the number of samples in each state group and their corresponding prediction accuracy, we can gain a more accurate assessment of model performance.

Process mining techniques, while intended to enhance NILM by identifying patterns between devices, did not yield useful results in this study. This could be due to incorrect setup, unsuitable parameters, or a lack of discernible patterns in the dataset.

Overall, while the Random Forest and RNN/LSTM models demonstrated strong potential for accurate device state prediction, the CNN model did not perform well. This underscores the importance of choosing the right model and parameters for NILM tasks, as well as the value of using comprehensive datasets that capture the full range of device interactions.

5.4 Limitations of the Results

The results of this study are limited to the specific types of devices chosen for analysis. This approach does not provide insights into devices with variable operating states or those that operate continuously, such as heating devices. Additionally, devices that are rarely turned on are not well-represented in the dataset, leading to a lack of predictive accuracy for these types of appliances.

Another significant limitation is the adaptability of the models to other households. Due to the lack of overlapping devices across the dataset, we were unable to test whether the trained models could generalize effectively to different households with varying appliance configurations.

Furthermore, due to the nature of some devices, certain operating states are not very common, resulting in insufficient data for those states. This scarcity of data leads to lower precision in their predictions, impacting the overall accuracy and reliability of the models.

5.5 Comparison with Objectives

The results partially met the objectives. While electrical devices were successfully identified, the application aspect of objectives remains incomplete. The original plan was to develop an application that allows users to monitor household electricity usage. This application would provide notifications about anomalies in the network, such as forgetting to close the refrigerator door or leaving the oven on for too long. However, this application has not been developed in the current scope of the study.

5.6 Avenues for Further Study

Future studies could focus on the following areas to further enhance the results and applicability of NILM methods:

- **Advanced Models for Load Disaggregation**:
  - Explore more advanced machine learning models, such as Transformer-based architectures, to improve load disaggregation results.

  - Further investigate how process mining techniques can be used to detect patterns and dependencies between different devices.

- **Development of a Household Energy Monitoring Application**:
  - Develop an application that leverages the created algorithms to provide real-time insights into household energy usage.
  - Start by testing the application in a controlled environment, then gradually increase the complexity and number of devices.
  - Implement a user interface that allows users to input data about device usage (e.g., indicating when a washing machine is turned on and for how long). This feedback will help the app learn and improve its predictions. Similar approach to Tibber 2.9.4.

- **Enhanced Feature Extraction**:
  - Extract additional features such as rapid rises in power consumption to gain further insights.
  - Use more advanced load monitors that provide additional information beyond active power to enhance feature extraction.

- **Disaggregation of Similar and Always-On Devices**:
  - Focus on disaggregating devices with very similar electricity usage patterns and those that are always on.
  - Address the unique challenges posed by overlapping usage patterns and continuous operation of certain devices.

5.7 Improvements for Future Work

This section discusses potential improvements to the methods used in this study:

- **RNN/LSTM Parameter Tuning**: Experimenting with different parameters in the RNN/LSTM method to improve performance. Adjusting window sizes and steps can create more training data, which is crucial for the learning process.

- **CNN Model Testing and Tuning**: Further testing and tuning of the CNN model, including remodeling or potentially skipping this method if it proves ineffective.
- **Larger and More Varied Datasets**: Using larger and more varied datasets to provide a more robust evaluation of the models.
- **Data Augmentation Techniques**: Incorporating data augmentation techniques to artificially expand the dataset and simulate real-world variability, helping to create a more robust model.
- **Better Validation Strategies**: Employing better validation strategies, such as cross-validation, to ensure that the models are tested thoroughly and reliably. This approach will help assess how well the models generalize to new, unseen data, reducing the risk of overfitting.

Chapter 6

Conclusions

This thesis aimed to develop reliable methods for identifying electrical devices in a household based on aggregate electrical signals. The primary objectives were to create or obtain a suitable dataset, implement the NILM procedure, and utilize machine learning methods to determine their effectiveness in learning device usage patterns. Additionally, explore the potential of these methods in practical applications for household energy management.

Research demonstrated that both Random Forest and RNN/LSTM models are effective in capturing the unique patterns of appliance usage in households. The Random Forest model achieved the highest overall accuracy, while the RNN/LSTM model showed strong potential in capturing temporal dependencies. In contrast, the CNN model did not perform well, likely due to the nature of the problem and the model's design.

The study highlighted the importance of using comprehensive datasets that include multiple devices simultaneously. This approach allowed the models to learn the interactions and overlapping usage patterns of different devices, leading to improved accuracy. Moreover, incorporating process mining techniques showed limited success in identifying patterns between devices, suggesting a need for further investigation.

However, several limitations were identified. The study did not address the adaptability of the models to different households, as the dataset lacked overlapping devices across houses. Additionally, devices with insufficient examples in the dataset were predicted with lower precision. These limitations highlight areas for improvement and further research.

Future work should focus on exploring more advanced models and using more diverse datasets to enhance model generalization. Developing a user-friendly application to provide real-time insights into household energy usage and incorporating user feedback could significantly improve the practical utility of NILM methods. Data augmentation techniques and better validation strategies, such as cross-validation, should also be employed to ensure robust model performance.

In conclusion, this research significantly advances NILM techniques by demonstrating the effectiveness of Random Forest and RNN/LSTM models and potential use of Process Mining in identifying household appliances. These findings address the motivation presented in the introduction by offering practical solutions for enhancing energy management in households. By developing methods that optimize electricity usage through precise device identification, detect unsupposed device usage to prevent hazards, and improve security and health monitoring by identifying irregular usage patterns, this thesis provides valuable tools for developing a functional NILM system.

Bibliography

[1] "Openai, chatgpt," https://www.openai.com/chatgpt, 2024, accessed: 2024-06-11.

[2] E. Aladesanmi and K. Folly, "Overview of non-intrusive load monitoring and identification techniques," *IFAC-PapersOnLine*, vol. 48, no. 30, pp. 415–420, 2015, 9th IFAC Symposium on Control of Power and Energy Systems CPES 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315030566

[3] G. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 198, accessed: 2024-05-01.

[4] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," *Proceedings of the 2011 SIAM International Conference on Data Mining*, pp. 747–758, 2011.

[5] M. Zeifman, "Disaggregation of home energy display data using probabilistic approach," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1, pp. 23–31, 2012.

[6] J. Kelly and W. J. Knottenbelt, "Neural NILM: deep neural networks applied to energy disaggregation," *CoRR*, vol. abs/1507.06594, 2015. [Online]. Available: http://arxiv.org/abs/1507.06594

[7] J. Z. Kolter and M. J. Johnson, "Redd: A public data set for energy disaggregation research," *Proceedings of the 2011 Workshop on Data Mining Applications in Sustainability (SIGKDD)*, pp. 1–6, 2011.

[8] D. Alonso, I. Etxeberria-Otadui, I. Rezabal, and I. Vidal, "Challenges in smart home energy disaggregation," *Energy Procedia*, vol. 62, pp. 192–200, 2014.

[9] N. Patel, P. Jain, M. Parashar, and R. Khanna, "Glimpse: A high-level programming language for the smart grid," *Proceedings of the 1st ACM International Conference on Energy-Efficient Computing and Networking*, pp. 1–6, 2010.

[10] D. Precioso Garcelán and D. Gomez-Ullate, "Nilm as a regression versus classification problem: the importance of thresholding," 10 2020.

[11] Akeptus, "Exploring the fundamentals of non-intrusive load monitoring (nilm) and its potential to improve energy efficiency in buildings | by akeptus | medium," https://medium.com/@akeptus/, 2024, accessed: 2024-05-12.

[12] A. M. Pirbazari, "Predictive analytics for maintaining power system stability in smart energy communities," pp. 24–25, 2021, accessed: 2024-05-03. [Online]. Available: https://uis.brage.unit.no/uis-xmlui/bitstream/handle/11250/2755369/Thesis_PhD_Aida_Pirbazari_open.pdf

[13] P. Nandy, "Hidden markov model based non-intrusive load monitoring using active and reactive power consumption," https://scholarsmine.mst.edu/cgi/viewcontent.cgi?params=/context/masters_theses/article/8612/&path_info=uc.pdf, 2016, accessed: 2024-05-10.

[14] "k-means clustering - wikipedia," https://en.wikipedia.org/wiki/K-means_clustering, 2024, accessed: 2024-05-13.

[15] I. Dabbura, "K-means clustering: Algorithm, applications, evaluation methods, and drawbacks | by imad dabbura | towards data science," https://towardsdatascience.com/, 2024, accessed: 2024-05-13.

[16] Y. Liu, C. Liu, Y. Shen, X. Zhao, S. Gao, and X. Huang, "Non-intrusive energy estimation using random forest based multi-label classification and integer linear programming," *Energy Reports*, vol. 7, pp. 283–291, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484721006478

[17] L. Numminen, "Process mining algorithms simply explained – workfellow," https://www.workfellow.ai/learn/process-mining-algorithms-simply-explained, September 5, 2023, accessed: 2024-05-27.

[18] V. Abeykoon, N. Kankanamdurage, A. Senevirathna, P. Ranaweera, and R. Udawapola, "Real time identification of electrical devices through power consumption pattern detection," 03 2016.

[19] J. Kelly and W. Knottenbelt, "Neural nilm: Deep neural networks applied to energy disaggregation," in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, ser. BuildSys '15. ACM, nov 2015. [Online]. Available: http://dx.doi.org/10.1145/2821650.2821672

[20] "nilmtk/nilmtk: Non-intrusive load monitoring toolkit (nilmtk)," https://github.com/nilmtk/nilmtk, 2024, accessed: 2024-06-01.

[21] "Et uvanlig strømselskap - tibber," https://tibber.com/no, 2024, accessed: 2024-05-14.

[22] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[23] T. pandas development team, "pandas-dev/pandas: Pandas," feb 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[25] F. Chollet *et al.* (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[26] "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[27] A. Berti, S. J. van Zelst, and W. M. P. van der Aalst, "Process mining for python (pm4py): Bridging the gap between process- and data science," *CoRR*, vol. abs/1905.06169, 2019. [Online]. Available: http://arxiv.org/abs/1905.06169

[28] N. Richardson, I. Cook, N. Crane, D. Dunnington, R. François, J. Keane, D. Moldovan-Grünfeld, J. Ooms, J. Wujciak-Jens, and Apache Arrow, *arrow: Integration to 'Apache' 'Arrow'*, 2024. [Online]. Available: https://github.com/apache/arrow/

[29] "Tutorial — hmmlearn documentation," https://hmmlearn.readthedocs.io/en/latest/tutorial.html, 2010, accessed: 2024-05-27.

[30] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[31] "datetime — basic date and time types — python 3.12.4 documentation," https://docs.python.org/3/library/datetime.html, 2024, accessed: 2024-06-02.

[32] J. Kelly and W. Knottenbelt, "The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes," *Scientific Data*, vol. 2, no. 1, mar 2015. [Online]. Available: http://dx.doi.org/10.1038/sdata.2015.7

[33] J. Z. Kolter and M. J. Johnson, "Redd: A public data set for energy disaggregation research," https://people.csail.mit.edu/mattjj/papers/kddsust2011.pdf, 2011, accessed: 2024-04-03.

[34] S. Makonin, "Ampds (the almanac of minutely power dataset)," http://ampds.org/, 2024, accessed: 2024-04-03.

[35] N. Batra, M. Gulati, A. Singh, and M. Srivastava, "It's different: Insights into home energy consumption in india," 11 2013.

[36] A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro, and A. M. Tonello, "Greend: An energy consumption dataset of households in italy and austria," 2014. [Online]. Available: http://arxiv.org/pdf/1405.3100

[37] C. Klemenjak, C. Kovatsch, M. Herold, and W. Elmenreich, "A synthetic energy dataset for non-intrusive load monitoring in households," *Scientific Data*, vol. 7, no. 1, apr 2020. [Online]. Available: http://dx.doi.org/10.1038/s41597-020-0434-6

[38] "ch-shin/awesome-nilm: A curated resources of awesome nilm resources," https://github.com/ch-shin/awesome-nilm, 2024, accessed: 2024-06-14.

[39] "Random forest algorithm in machine learning - geeksforgeeks," https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/, 2024, accessed: 2024-06-02.

[40] "One hot encoding in machine learning - geeksforgeeks," https://www.geeksforgeeks.org/ml-one-hot-encoding/?ref=header_search, 2024, accessed: 2024-06-14.

[41] K. Eckhardt, "Choosing the right hyperparameters for a simple lstm using keras | by karsten eckhardt | towards data science," https://towardsdatascience.com/, 2024, accessed: 2024-06-14.

[42] "How to choose cnn architecture mnist," https://www.kaggle.com/code/cdeotte/how-to-choose-cnn-architecture-mnist, 2024, accessed: 2024-06-14.

University of Stavanger

4036 Stavanger
Tel: +47 51 83 10 00
E-mail: post@uis.no
www.uis.no