



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelor Thesis

Amrit Raj

Energy Disaggregation using Deep Neural
Networks on Household Appliances

Amrit Raj

Energy Disaggregation using Deep Neural
Networks on Household Appliances

Bachelor Thesis
is based on the study regulations
for the Bachelor Thesis of Engineering degree programme
Information Engineering
at the Department of Information and Electrical Engineering
of the Faculty of Engineering and Computer Science
of the Hamburg University of Applied Sciences

Supervising examiner : Prof. Dr. rer. nat. Wolfgang Renz
Second Examiner : Prof. Dr.Ing. Sebastian Rohjans

Day of delivery: 3 July 2017

Amrit Raj

Title of the Bachelor Thesis

Energy Disaggregation using Deep Neural Networks on Household Appliances

Keywords

Energy Disaggregation, Deep Neural Networks, Multi-Layer Perceptron, Convolutional Neural Networks

Abstract

Diverse deep neural network (DNN) approaches have displayed high accuracy in the fields of pattern recognition and image classification but their potential has not been explored in the field of energy disaggregation. The aim of this thesis is to investigate the accuracy with which two DNN approaches classify active household appliances for energy disaggregation and compare the performance of DNN with other classification methods used in the field.

The first approach used is the Multi-Layer Perceptron (MLP) approach which is one of the simplest DNN methods and it displays baseline accuracy of any DNN. The second approach is Convolutional Neural Networks (CNN), which is more advanced and improves upon the baseline accuracy. Both approaches are tested with various optimizers, activation functions and loss functions as performance measures. Open source data (REDD dataset) is used to train and test the neural networks. The dataset consists of 6 houses which are used for training and the testing 3 labelled appliances common among the houses (dishwasher, lighting and washer dryer) with the addition of unknown appliance data. When presented with real world data which included unknown devices in addition to the three labelled devices, CNN achieved an accuracy of 90.00% and MLP achieved an accuracy of 77.14%. When the experiment was repeated with data including only the known devices, CNN achieved an accuracy of 95.83% and MLP achieved 80.37%.

Contents

List of Tables	6
List of Figures	7
1 Introduction	9
1.1 Motivation	9
1.2 Problem Statement	10
1.3 Thesis Overview	10
2 Literature Survey	12
2.1 Energy Disaggregation	12
2.2 Related Work	14
2.3 Machine Learning	17
2.3.1 Supervised Learning	17
2.3.2 Classification	18
2.3.3 Deep Learning	18
2.4 Multi-Layer Perceptron	23
2.5 Convolutional Neural Network	25
3 Concepts	30
3.1 Dataset	30
3.1.1 Overview	30
3.1.2 Analysis	34
3.1.3 Windowing Data	36
3.1.4 Data preprocessing	38
3.2 Appliances Detected	40
3.3 Test Structure	41
3.4 Design Flow	45
4 Implementation of Multi-Layer Perceptron	46
4.1 Implementation Design	46
4.2 Results	48

5	Implementation of Convolutional Neural Network	50
5.1	Implementation Design	50
5.2	Results	52
6	Evaluation	54
7	Conclusion and Future Work	60
8	Appendix A	63
9	Appendix B	66
10	Appendix C	68
	Bibliography	69

List of Tables

2.1	Percentage of total energy classified correctly in [51]	15
2.2	Percentage of energy correctly classified in [50]	16
2.3	Percentage of energy correctly classified in [41]	16
2.4	List of activation functions available in Keras	21
3.1	REDD data description	32
3.2	Analysis of sampling period of 3s in "REDD_low_freq" data	34
3.3	Overview of common Appliance Data	43
3.4	Overview of Training and Testing Data for common Appliances	44
3.5	Overview of common Appliance Data with unknown appliances	44
3.6	Overview of Training and Testing Data for common and unknown appliances	44
4.1	The effect of window size and loss function on validation accuracy for MLP architecture	48
4.2	The effect of window size and loss function on validation accuracy for MLP architecture with unknown appliances	49
5.1	The effect of window size and loss function on validation accuracy for CNN architecture	52
5.2	The effect of window size and loss function on validation accuracy for CNN architecture with unknown appliances	53
6.1	Highest accuracy of each architecture for MLP and CNN networks from Table 4.1 and Table 5.1	57
6.2	Comparison of results with related work	59
8.1	Complete data description of REDD_low_freq folder	63
8.2	Number of samples that exceed 3 s interval for each house	64

List of Figures

2.1	Energy disaggregation overview [27]	12
2.2	Energy disaggregation popularity	14
2.3	Supervised learning	18
2.4	Biological inspiration for perceptron [90]	19
2.5	Derivatives of function used to find minimum by gradient descent [25]	23
2.6	Multilayer perceptron architecture [38]	24
2.7	Architecture of typical CNN [25]	27
2.8	Architecture of CNN used in [61]	28
2.9	Architecture of CNN used in [52]	28
3.1	REDD data set structure	31
3.2	Power consumption of dishwasher from house 1	33
3.3	A complete cycle of dishwasher consumption from house 1	33
3.4	Measurement samples from dishwasher	35
3.5	Hardware and software setup for REDD dataset [51]	36
3.6	Window size visualization for lighting appliance from House 1	38
3.7	Raw data from dishwasher from House 1	39
3.8	Outlets unknown appliance readings for 3 day window size	41
3.9	Differing Power Consumption of Lighting and Washer Dryer appliances	43
3.10	Overview of design flow	45
4.1	Basic design architecture for MLP	47
5.1	Basic design architecture for CNN	51
6.1	Behaviour of overfitt, underfitt and optimal capacity [25]	54
6.2	Validation loss (error) of MLP over epochs for 3 day window size and 8 neurons per layer architecture	55
6.3	Validation loss (error) of CNN over epochs for 3 day window size and 8 neurons per layer architecture	55
7.1	Overview of Lambda architecture	61
8.1	Measurment samples from lighting	64

8.2	Measurment samples from washer dryer	65
9.1	Optimizer effect on loss when window size is 2 hr for 20 epochs	66
9.2	Effect of Optimizer on test accuracy for MLP approach of window size 2 hr	67
10.1	Folder structure in provided CD	68

1 Introduction

1.1 Motivation

With the rise in demand for sustainable energy sources, it becomes a priority to develop a grid infrastructure which is reliable in providing energy consistently, flexible in handling changing power demands and highly efficient in mitigating energy losses. Maximizing the efficient usage of resources and available power enables the development of a sustainable balance between supply and demand.

Much research [17] [19] [12] has been carried out to investigate the potential of saving household electricity when energy consumption is disaggregated per appliance. For the consumer, being aware of the energy breakdown per appliance encourages change in usage patterns of such appliances to reduce unnecessary costs. For public policy making bodies, such a disaggregation allows the identification of appliances which consume needless amounts of power and lead to avoidable negative impacts on the environment, and they can thereafter make suitable policies. For the industry, this serves as motivation to modify and improve appliance design [21]. Moreover, one such study Karen Ehrhardt-Martinez et al. [17] even determined that if by 2030 well designed programs which disaggregated energy consumption on appliance level are fully integrated throughout the residential sector, they can provide the equivalent of 100 billion kilowatt-hours of electricity savings on an annual basis in the USA alone. Thus, highlighting the enormous scale of financial advantage such programs can yield. Finally, understanding the per appliance breakdown of energy consumption also allows energy providers to make more accurate consumption forecasts.

However, disaggregating energy consumption to accurately classify appliances is challenging. Traditional approaches such as Hidden Markov Chain modeling and Signal Processing are not able to accurately address situations where different appliances have similar power consumption, similar appliances have different power consumption and when an appliance has continuously varying power consumption.

The aim of this thesis is to accurately classify household appliances in a small interval (window) from the power consumption data of the household appliances. This is beneficial, as a consumer can be notified of how their power is being consumed almost immediately. The interested party could access an overview of their power consumption from previous weeks

and adjust their usage accordingly to save more money while being environmentally friendly. Furthermore, governments can use this information for adjusting their energy consumption goals each year in a more efficient manner. The Paris Agreement, which has been signed by 195 countries aims to combat climate change by using new technology framework to reduce their emissions [70]. Reducing these emissions can be achieved by making more efficient usage of common appliances by citizens of the 195 countries.

In this thesis, Deep Neural Networks (DNN) are used to classify household appliances from large amounts of energy profile data. The decision for using DNNs was influenced by its success in ongoing research in the fields of image recognition and speech recognition, which reflects the increased accuracy of using DNNs for pattern recognition. Implementing DNN for image recognition resulted in 1st and 2nd place prizes in the ImageNet challenge [83] for researchers from University of Oxford. Moreover, research done by Microsoft [30] in speech recognition also used DNN for automatic speech recognition and emphasized that three major speech research groups (Google voice, Youtube speech, English broadcast news) achieved significant improvements in a variety of state-of-the art automatic speech recognition systems by replacing Gaussian mixture models and Hidden Markov models with DNNs.

1.2 Problem Statement

The aim of this thesis is to accurately and efficiently classify household appliances in small time intervals (window) from the power consumption data of household appliances using deep neural networks. The data which will be used for the household appliances to train the neural network models is from the open source Reference Energy Disaggregation Dataset (REDD). While this thesis focuses on REDD dataset, the implementation provided can be replicated to other datasets for energy disaggregation as the approach remains the same. This thesis aims to implement Multi-Layer Perceptron (MLP) to determine baseline accuracy and thereafter, through researching relevant literature, determine and implement an approach which further optimizes the accuracy.

1.3 Thesis Overview

Chapter 2: Literature Survey explores literature relevant to Energy Disaggregation, Machine learning and Deep Learning.

Chapter 3: Concepts explores the concept of this thesis's methodology by making the necessary design decisions including the data pre-processing, test structure and design flow.

Chapter 4: Implementation of Multi-layer Perceptron aims to describe the implementation of MLP architecture and provides the results obtained from this approach.

Chapter 5: Implementation of Convolutional Neural Network aims to describe the implementation of CNN architecture and provides the results obtained from this approach.

Chapter 6: Evaluation includes the discussion for MLP and CNN results and the comparison between both approaches.

Chapter 7: Conclusion and Future Work presents a conclusion to this thesis and proposes relevant future research work.

Appendices: Appendix A and B include the supporting information in reference to work from previous chapters. Appendix C includes information regarding the CD structure.

2 Literature Survey

2.1 Energy Disaggregation

Energy disaggregation is the act of identifying individual appliance signatures from a total power consumption reading [27]. Alternatively, it is also called Non-Intrusive Load Monitoring (NILM), Non-Intrusive Appliance Load Monitoring (NIALM) and Nonintrusive Appliance Load Monitoring (NALM) in different papers. Figure 2.1 shows the recorded power consumption of a sample house and implements the concept of energy disaggregation by identifying the different signal signatures of common appliances.

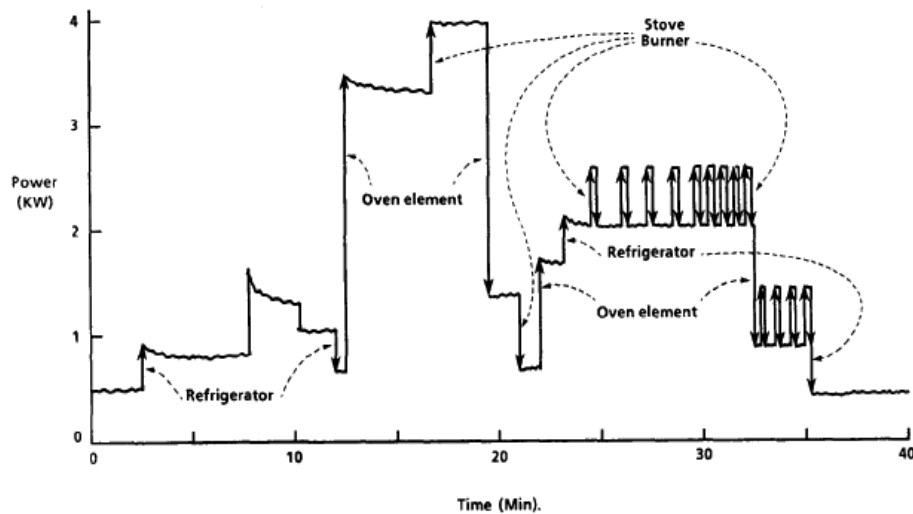


Figure 2.1: Energy disaggregation overview [27]

Energy disaggregation can be used for a range of purposes and by many sectors including residential, commercial and industrial. The approach taken in this paper can be replicated for all the sectors. The majority of research for energy disaggregation uses open source residential data for analysis as the findings can be published and discussed, while commercial and industrial data remain to be confidential.

This topic was initially envisioned at MIT by G.W Hart in 1980s [27]. In his paper he introduces energy disaggregation and proposes an approach to collect and analyze appliance data from households. He also suggests prospective applications and associated use cases. This paper focuses on providing a different approach for analyzing the data. Nonetheless, with ongoing research in the Internet of Things (IoT) [4] field, collection of data can be done differently to be more efficient and resourceful. The European Commission has interest [98] in promoting the usage of such devices and there is already a noticeable increase in the number of smart appliances sold across Europe. These devices will make consumption energy data available to the consumers, thereby helping to make the collection of data simpler and more efficient.

With an earlier approach for collecting data, G.W Hart created a device which was installed at the revenue-meter socket of a residence which used signal analysis techniques on the voltage and current waveforms to detect the usage characteristics of the individual appliances within the home that constitute the load. The approach taken involved monitoring the changes in the on/off state of an appliance within the residence so that changes in the total power consumption could be seen. While this is successful for simple on/off appliances, it is not applicable to complex appliances with many states such as dishwashers (as illustrated in Figure 3.3) and washing machines. The author suggests 3 different appliance models namely ON/OFF, Finite state machine and Continuously variable appliances but only tests and implements the ON/OFF model.

Much of the research in this field continues to use signal processing to improve the accuracy of previous models. According to this paper [102] which reviews the methodology of popular approaches for energy disaggregation, there is no complete solution suitable for all types of household appliances. Also, no complete set of robust, widely accepted appliance features has been identified. This thesis aims to tackle this issue and provide a more flexible approach.

Figure 2.2 illustrates the increase in related research being done for energy disaggregation. The number of publications are the number of published papers that cite [27]. These results were obtained using the Google scholar search engine. After the year of 2014, around 200 publications are made each year. This helps to portray growing interest and potential for energy disaggregation in the near future.

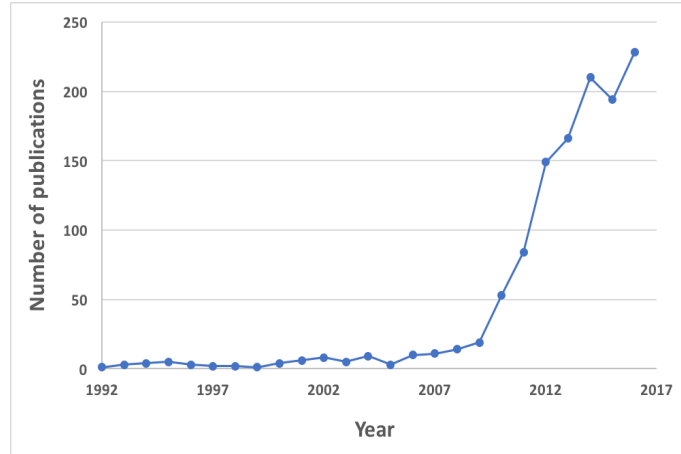


Figure 2.2: Energy disaggregation popularity

The advantage of using Deep neural networks is that the state of the appliance does not matter in the formulation of the model. DNNs are artificial neural networks (ANN) that have multiple layers in their network. Deep learning introduces additional explicit and implicit learning priors in order to reduce the generalization error compared to traditional machine learning techniques. Therefore, being very useful for pattern recognition as an unknown input or output can be mapped to a corresponding known input or output. ANNs are useful because of their adaptability and providing evidential response [75] by returning a decision with a measure of confidence. This allows the researcher to evaluate the reliability and accuracy of their model.

Multi-layer perceptron (MLP) and convolutional neural networks (CNN) are the types of DNN that are being investigated. Both approaches are explained in detail in their respective chapters.

2.2 Related Work

By being one of the earliest publicly available data sets REDD attracted many different researchers and approaches. The majority of approaches use Signal processing, Machine learning and Deep Learning. Signal processing techniques have traditionally been used, while machine learning and Deep Learning are more recent approaches.

The authors of the REDD data in their study **REDD: A Public Data Set for Energy Disaggregation Research** [51] set implemented Factorial Hidden Markov Model on their data. FHMM[22] are a generalization of HMM in which the hidden state is factored into multiple state variables and is therefore represented in a distributed manner. Using this approach,

they attempted to predict the behaviour of an appliance and then compared their prediction to the actual signature. The performance measure used in this study is the total energy correctly classified. The authors use individual appliance energy sequences to train the HMMs using the standard Baum-Welch algorithm [67]. They use 4 states per device and typically 20 devices per home which results in 1×10^{12} different combinations of hidden states. To evaluate their approach the authors use **two weeks** of data from 5 houses and sub-sampled the data to 10 second intervals as a preprocessing step. The appliances used for the training are provided in the appendix. The accuracy of their approach is illustrated in the Table 2.1. The average accuracy of their approach for training is 64.5%, while when testing is 47.7%.

House	FHMM	
	Train	Test
1	71.5%	46.6%
2	59.6%	50.8%
3	59.6%	33.3%
4	69.0%	52.0%
6	62.9%	55.7%
Total	64.5%	47.7%

Table 2.1: Percentage of total energy classified correctly in [51]

In the study **Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation** [50] one of the authors from REDD dataset proposes an Additive Factorial Approximate MAP (AFAMAP) algorithm which is an unsupervised learning approach in continuation the FHMM approach. For the training data the author observed power output, and transitions probabilities set based upon the amount of time spent at each power level for an appliance and looked at all pairwise probabilities between them using the k-nearest-neighbor graph. This resulted in selecting 9 distinct appliances. For all methods, regularization were fit using one day of the data. For testing they report precision and recall metrics at the circuit level. Recall measures what portion of a given circuit's energy is correctly classified, while precision measures, of the energy assigned to a circuit, how much truly belonged to that circuit. The results of the approach are available in Table 2.2 where, performance is reported as precision/recall and bold entries denote statically significant better performance on both metrics in the table. These results are for **two weeks** of data. Comparatively it performs much better than the previous study. However, the previous study tests all appliances across 5 houses whereas this study tests only 7 appliances which can be part of the same circuit/house. Comparatively, the first study has broader test scenarios, which increases the reliability of the result. Nonetheless, the recall values of the study average to 60.3% which makes it 23% higher than the previous studies tested approach.

Circuit	AFAMAP
1 Microwave	97.5% / 66.1%
2 Bath GFI	82.7% / 70.8%
3 Electronics	41.6% / 0.8%
4 Kitch. Out. 2	37.5% / 12.9%
5 Furnace	91.7% / 70.8%
6 Kitchen. Out. 2	45.2% / 16.0%
7 Wash/Dryer 2	98.8% / 73.6%
Total	87.2% / 60.3%

Table 2.2: Percentage of energy correctly classified in [50]

In the study **Bayesian Nonparametric Hidden Semi-Markov Models** [41] the authors use an explicit-duration Hierarchical Dirichlet Process Hidden semi Markov Model (HDP-HSMM) [40]. HDP-HSMM is a natural Bayesian non-parametric extension of the traditional HMM. The methods introduced also provide new methods for sampling inference in the finite Bayesian HSMM. The authors use the REDD dataset to test their approach. The authors chose the top 5 power-drawing devices (refrigerator, lighting, dishwasher, microwave, furnace) across several houses and identified 18 24-hour segments across 4 houses for which many (but not always all) of the devices switched on at least once. To which they applied a 20-second median filter which resulted in each sequence being approximately 5000 samples long. The authors do not mention if they used the high frequency or low frequency data. However, since there measurements are across 4 houses, this is only possible for the low freq data and thus the 5000 samples corresponds to ≈ 4.16 hours. To measure performance the same metrics are used as in [51]. The results of this approach are illustrated in Table 2.3. The results display an average accuracy of 47.7% for the least accurate and 81.5% for the most accurate approach respectively.

House	EM-FHMM	F-HDP-HMM	F-HDP-HSMM
1	46.6%	69.0%	82.1%
2	50.8%	70.7%	84.8%
3	33.3%	67.3%	81.5%
6	55.7%	61.8%	77.7%
Mean	47.7%	67.2%	81.5%

Table 2.3: Percentage of energy correctly classified in [41]

According to the study **Non-Intrusive Appliance Load Monitoring (NIALM): Review and Outlook**[102] most researchers agree that in order to reach high accuracy of detection of

appliances the microscopic and macroscopic features of the electric signal should be used. In order to capture the microscopic features a minimum rate of 1.2 kHz- 2 kHz is needed. Comparatively, this paper uses sampled data at 1/3 Hz which illustrates that with a deep neural network approach, less data can be used to make even more accurate predictions.

2.3 Machine Learning

Machine learning is the science of getting computers to act without being explicitly programmed. Machine learning consists of the following approaches: Supervised, Unsupervised and Reinforcement learning. Supervised learning is the learning to infer a function from labeled training data. Unsupervised learning is the learning to infer a function from unlabeled training data and Reinforcement learning is the learning based on feedback or reward. This thesis implements supervised learning as the REDD dataset includes labeled appliance data. As the scope of the thesis is to detect appliances, the task of identifying appliances in houses naturally becomes a supervised classification task. However, an unsupervised approach can also be taken for future work to predict an appliance's power consumption in the near future. In a usual use case unsupervised learning will be implemented when a lot more data is available for unknown appliances, and the task is to try to label the unknown appliances [58].

Common problem solving tasks where machine learning is used are anomaly detection, classification, clustering, regression and rule extraction. This thesis treats the problem as a classification problem as the goal is to predict an appliance in use. These appliances are treated as classes.

2.3.1 Supervised Learning

The input variable is denoted as input **features**, while the output variable is denoted as **target**. A pair of input features and target variable is called a **training set**. A **validation set** can be taken out of the training set for testing as long as it not used while the model is training. A validation set is usually 10-20% of the entire data-set. The aim of supervised learning is to learn a function $h : X \mapsto Y$ for a given data-set so that $h(x)$ is an accurate predictor for the corresponding value of y . The function h is called a hypothesis [71].

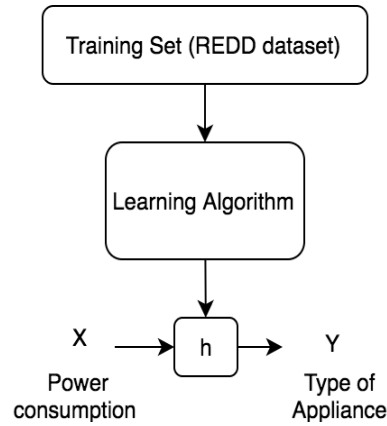


Figure 2.3: Supervised learning

2.3.2 Classification

Classification is a type of task, in which a learning algorithm is tasked to specify which of the ' n ' categories some input belongs too. The learning algorithm produces a function $f : \mathbb{R}^n \rightarrow \{1, \dots, k\}$. The model assigns an input described by a vector x to a category identified numerically as y , usually a '1' for positive identification else '0'.

2.3.3 Deep Learning

Multiple definitions for Deep Learning can be found. Nonetheless, a holistic summary is that Deep Learning is a class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification [14]. Deep learning is in the intersections among the research areas of neural networks, artificial intelligence, graphical modeling, optimization, pattern recognition, and signal processing. Deep learning differs from traditional machine learning approaches such as decision tree [57], bayesian methods [56] etc by aiming to learn feature hierarchies. Automatically learning features at multiple levels of abstraction allow a system to learn complex functions mapping the input to the output directly from data, without depending completely on human-crafted features. This is especially important for higher-level abstractions, which humans often do not know how to specify explicitly in terms of raw sensory input [6]. The ability to automatically learn powerful features, especially in the context of this thesis, will become increasingly important as the amount of data and range of applications continues to grow. The following subsections provide the core concepts in deep learning.

Perceptron

Perceptrons were examples of statistical pattern recognition systems and the first artificial neural networks. This concept was introduced by Rosenblatt [78]. He worked on the model introduced by Warren McCulloch and Walter Pitts [65] in 1943 where they contended that neurons with a binary threshold activation function were analogous to first order logic sentences [91]. He was also inspired by the work of Donald Hebb which later became referred to as Hebb's rule. Hebb's rule states that "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased"[29]. Rosenblatt's model of a perceptron was learning in the "Hebbian" sense, through the weighting of inputs. Figure 2.4 illustrates the mathematical model for the perceptron and its biological inspiration.

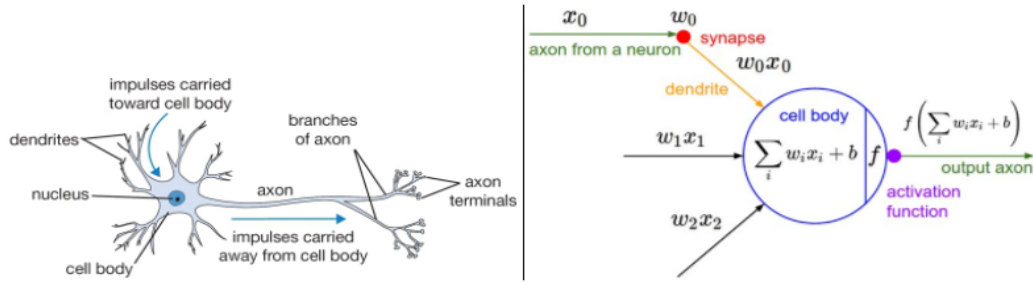


Figure 2.4: Biological inspiration for perceptron [90]

In Keras the dense layer [44] implements the operation illustrated in Figure 2.4 where $\text{output} = \text{activation}(\text{dot}(\text{input}, \text{kernel}) + \text{bias})$, here the `kernel` and `bias` are taken as default values. The `kernel(weights)` default value is a tensors with a uniform distribution from -0.05 to 0.05 [45]. This method of initialization is used in other neural network libraries such as Tensorflow and Theano and is based on the findings of this research [23]. The method ensures that weights in a network are neither too small or too large. If the weights are too small the output signal can reduce further as it passes through each layer until it is insignificant. If the weights in a network start too large, then the output signal grows as it passes through each layer, becoming too large to be useful. The `bias` is an optional value and is not implemented in the model for this thesis as the a bias unit is just appended to the start/end of the input and each hidden layer, and isn't influenced by the values in the previous layer. The activation function is a parameter which is investigated and reported in the results section.

Feed forward

In feed forward networks information flows from the left to the right of the model as in Figure 2.6. The input features x are used to compute the responses of the first layer through an activation function. These computed values are then fed in to the next hidden layer as inputs and passed through another activation function till the output layer is reached.

Activation function

The activation function defines the output of a node given an input or set of inputs as illustrates in Figure 2.4. The activation function used by McCulloch and Pitts was the threshold step function. Other activation functions that are widely used are the Sigmoid, Piecewise Linear, ReLu and Softmax functions, etc. For the purpose of this thesis all activation functions are tested which are provided in Keras. Among the activation functions ReLu and Softmax are the most popular. ReLu is popular as the function is one of the most widely used and it has proven to be faster and more efficient for large neural networks due to its linear nature [83]. The following Table 2.4 illustrates a summary of some available activation functions in Keras. All the activation functions are available in the source library [46]. These activation functions can be grouped into 3 categories, namely threshold function, linear activation or non linear activation function. ReLu is an example for a linear activation function and Sigmoid, Softsign are examples of non-linear activation functions. Non-Linear activation functions are differentiable, continuous and monotonically increasing. Introducing non-linearity extends the kinds of functions that we can represent with our neural network.

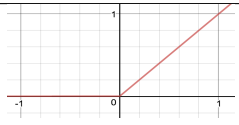
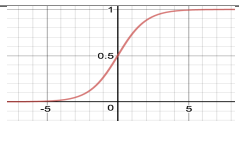
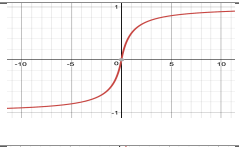
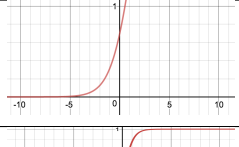
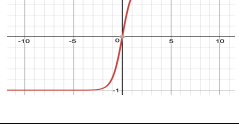
Activation Function	Equation	Figure	Range
Rectified linear unit (ReLU)	$f(x) = \max(0, x)$		[0,1)
Sigmoid	$f(x) = \frac{1}{1 + \exp(-x)}$		(0,1)
Softsign	$f(x) = \frac{x}{1 + x }$		(-1,1)
Softplus	$f(x) = \ln(1 + \exp(x))$		(0,∞)
Tanh	$f(x) = \frac{2}{1 + \exp(-2x)} - 1$		(-1,1)

Table 2.4: List of activation functions available in Keras

Loss functions

In neural network terminology an **epoch** is one forward pass and one backward pass of all the training examples.

Loss functions, also known as cost functions, are used to measure the degree of fit in the neural network. One way of measuring the performance of the model is to compute the **mean squared error** (MSE) of the model on the test set. MSE is used for linear regression algorithm [25]. In the case of linear regression the assumption is that the output $y \in \mathbb{R}$ is a linear function of the input. The MSE is measured per sample of an epoch and is defined as

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$

where \hat{Y}_i is the prediction vector, Y_i is the corresponding output vector and n is the total number of classes. As an example, in the case of 3 class classification one \hat{Y}_i vector will consist of 3 values corresponding to the 3 classes. The implication of using MSE is that larger errors are given more importance than smaller ones. There are some disadvantages to this simple method. Firstly, MSE is sensitive to extreme values. Secondly, Chatfield [7] and Armstrong

et al.[3] correctly identified that MSE calculates absolute measures and is scale dependent, so it becomes difficult to make comparisons between different series. Nonetheless, MSE produces a fast computation for the neural network[55].

Another loss function used for measuring performance in this thesis is the **categorical cross-entropy** (CCE). It is the choice for multi-class classification problems and softmax activation output units [2] [24]. It is used for multi-class classification problems such as for the MNIST dataset and has been used and provided officially in the Keras examples [77]. MNIST dataset consists of handwritten digits from 0-9 and the models use CCE for measuring the performance of the network for predicting the 10 classes [62].

CCE is provided in the Lasagne library [88] which Keras uses in the backend [63] as:

$$CCE_i = - \sum_j t_{i,j} \log(p_{i,j})$$

where p is the prediction vector in the range (0,1) due to the softmax activation function used in the last layer. t is the target vector which is in the same format as that of the p vector. The index i represents each sample and j represents the number of classes. The CCE is also sensitive to extreme values but to lesser extent than MSE [76].

Backpropagation

The learning procedure in a neural network determines the internal parameters of the hidden units based on its knowledge of the inputs and desired outputs. This is achieved by having a forward pass for each input-output case in the neural network, to compute the activity levels of all neurons in the network. Then, a backward pass starting at the output neurons to compute the error derivative, thereby repeatedly adjusting the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector [100]. This concept is referred to as back-propagation which is a form of gradient descent.

Optimization

Optimizers are used to minimize loss function. Optimization refers to the task of either minimizing or maximizing some function $f(x)$ by altering x . Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks [79].

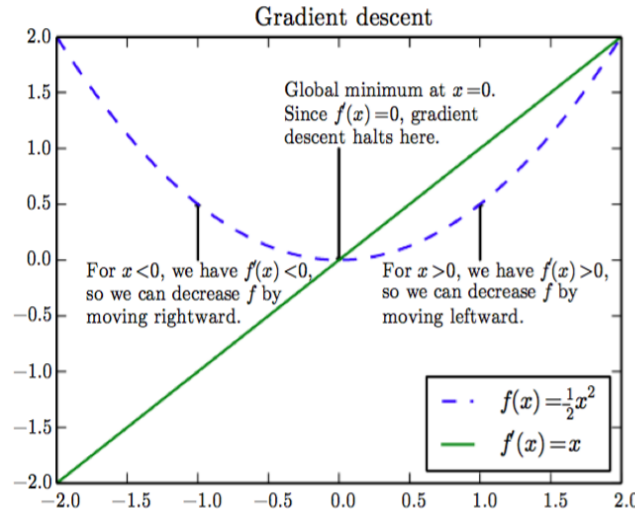


Figure 2.5: Derivatives of function used to find minimum by gradient descent [25]

Keras contains implementations of various algorithms to optimize gradient descent. The various optimizers available is found here [48]. An illustration of gradient descent is provided in Figure 2.5. Taking the derivative is therefore useful for minimizing a function because it tells us how to change x in order to make a small improvement in y .

In this thesis, the best suited optimizer is used for the neural network model. This is achieved by testing all the optimizers provided in the Keras library and choosing the optimizer that achieves the lowest error in 20 epochs. This is illustrated in the Appendix in Figure 9.1 where the left subplot illustrates the performance of all the optimizers while the right subplot illustrates the Adadelta optimizer which reaches the lowest error and highest accuracy in 20 epochs. It is important to note that most optimizers achieved a similar accuracy at the 20th epoch, however Adadelta performed better and was chosen to model the neural networks. The highest accuracy is illustrated in Appendix Figure 9.2. A detailed explanation of the window size is provided in the next section where the problem statement is addressed.

2.4 Multi-Layer Perceptron

A multilayer perceptron (MLP) is a feedforward artificial neural network model that maps sets of input data onto a set of appropriate outputs. An MLP consists of multiple layers of simple two-state, sigmoid processing elements (nodes) that interact using weighted connections. [73]. After an input layer, there are usually a number of intermediate (or hidden) layers

followed by an output layer on top. All neurons in a layer are fully connected to neurons in adjacent layers as illustrated in Figure 2.6 .

A standard MLP feedforward network with as few as a single hidden layer and arbitrary bounded and non constant activation function are found to be universal approximation functions for an arbitrary finite input environment measures, provided sufficiently many hidden units are available [32]. As concluded in the study, the results do not mean that all activation functions will perform equally well in specific learning problems. In applications, additional issues for example, minimal redundancy or computational efficiency, have to be taken into account.

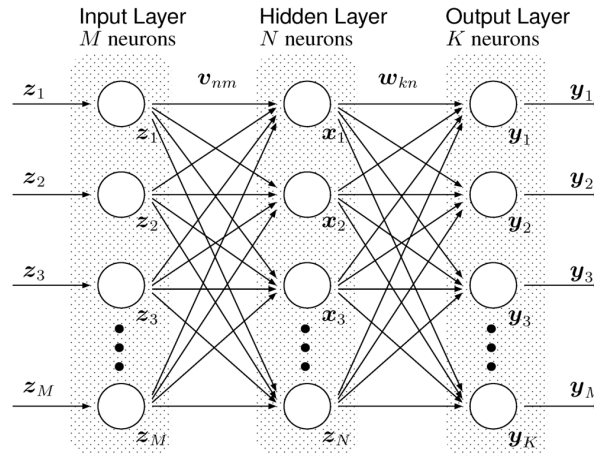


Figure 2.6: Multilayer perceptron architecture [38]

Advantages

Neural networks do not make any assumption regarding the underlying probability density functions or other probabilistic information about the pattern classes under consideration in comparison to other probability based models [54]. Furthermore, a two layer backpropagation network with sufficient hidden nodes has been proven to be a universal approximator [33]. Lastly, MLP are fast to implement and require lower CPU utilization when compared to other Deep Learning approaches (CNN, RNN etc).

Challenges

There are two central challenges being faced when using MLP network. Overfitting and underfitting. Underfitting occurs when a model is not able to obtain a sufficiently low error value on the training set. While, overfitting occurs when the gap between the training error

and test error is too large. In order to deal with these challenges, regularization is used in the neural network. Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.

It is not guaranteed that the training algorithm will be able to learn that function. Even if the MLP is able to represent the function, learning can fail for a number of reasons. The optimization algorithm used for training, may not be able to find the value of the parameters that corresponds to the desired function. Another reason is that the training algorithm might choose the wrong function due to overfitting. Feedforward networks provide a universal system for representing functions [25], in the sense that, given a function, there exists a feedforward network that approximates the function. The no free lunch theorem [101] states that there is no universal procedure for examining a training set of specific examples and choosing a function that will generalize to points not in the training set. Hence, we approach this problem by building a set of preferences into the learning algorithm. When these preferences are aligned with the learning problems, it performs better. The end goal is to try to reduce the training error and make minimize the gap between training and test error.

The ability to perform well on previously unobserved inputs is called generalization [25]. Steps taken to better generalization by reducing challenges of underfitting and overfitting are dropout for MLP and CNN, maxpool for CNN.

2.5 Convolutional Neural Network

Convolution neural networks (CNNs) are a type of feed-forward artificial neural network (ANN). Convolutional networks take inspiration from biological processes are directly inspired by the classic notions of simple cells and complex cells in visual neuroscience [60]. They have wide applications in image and video recognition, recommendation systems [97] natural language processing [9].

Compared to MLP, CNN leverage three important ideas that can help to improve the machine learning system. Namely, **sparse interactions**, **parameter sharing** and **equivariant representations** [25]. **Sparse interaction** help to store fewer parameters which reduces the memory requirements of the model and improves its statistical efficiency. This also means that computing the output requires fewer operations. This is achieved by making the kernel (matrix convolved with) smaller than the input. **Parameter sharing** is useful because rather than learning a separate set of parameters for every location, only one set is learned. This does not effect the run time but it further reduces the storage requirements of the model. And **equivariance** is a form of parameter sharing, where if the input changes, the output changes in the same way: $f(g(x)) = g(f(x))$. This is useful as for when it is known that that some

function of a small number of readings in data is useful when applied to multiple input locations. For example, The same pattern or cycle of an appliance appears quite frequently in the data series, so it is practical to share parameters across the entire dataset.

The convolution operation in 1D is defined as:

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\tau) \cdot g(x - \tau) d\tau$$

In essence, the convolution of $f(x)$ with $g(x)$ produces a third function. The operation consists of one of the original functions, giving the integral of the pointwise multiplication of the two functions as a function of the amount that one of the original functions is translated. The forward pass of the convolutional layer can in each depth slice be computed as a convolution of the neuron's weights with the input volume. Hence the name: Convolutional Layer [1].

For this thesis, 2D convolution is applied as it is just extension of previous 1D convolution by convolving both horizontal and vertical directions in 2 dimensional spatial domain. Supervised learning is a high-dimensional interpolation problem [64]. Hence, representing the data set in 2D is better suited for the convolutional neural architecture.

The key elements in the convolution neural architecture are the following: **convolutional layer, pooling layer, fully-Connected Layer**. Figure 2.7 illustrates the components of a typical convolutional neural network.

The **convolutional layer** will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. In Keras, the convolutional layer is referenced as a Conv2D and this layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs [47].

The **pooling layer** will perform a down sampling operation along the spatial dimensions (width, height). This allows to help with the problem of over-fitting by providing an abstracted form of the representation and also reduces the computational cost by reducing the number of parameters to learn.

The **fully connected layer** will compute the class scores. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume

CNN are among the most popular neural network that have been implemented in the Deep Learning field. Hence, significant work has been done related to advance the research in CNN. The following papers have contributed to the ongoing development of CNN.

Gradient Based Learning Applied to Document Recognition [61]: One of the earliest papers on CNN. The paper discovered that CNN were the most successful approach for

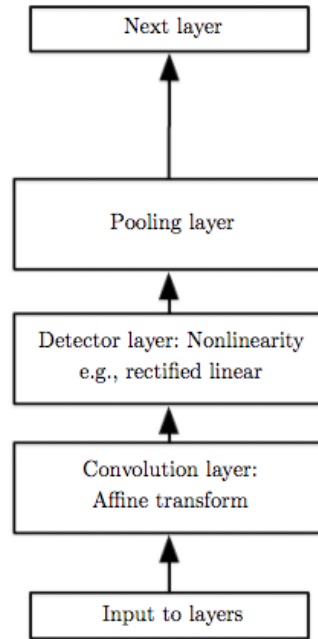


Figure 2.7: Architecture of typical CNN [25]

identifying hand written digits at the time compared to Linear classifier, K-nearest neighbour, Multi-layer perceptron, etc. The paper suggested that simultaneous automatic learning of segmentation and recognition can be achieved with gradient based learning methods. The paper states that CNN have been shown to eliminate the need for hand crafted feature extraction. Thereby, providing a promising tool for improving the previous approaches of signal processing to energy disaggregation. Most importantly, the paper introduced LeNet-5 architecture for CNN which is still widely used and is illustrated in Figure 2.8. Interestingly, the paper successfully predicted in 1998 that as training data gets plentiful, and computers get faster, our understanding of learning algorithms improve, recognition systems will rely more and more on learning, and their performance will improve. With the advancement of cheaper GPUs the trend of Deep Learning has significantly increase in the past years.

Imagenet classification with deep convolutional neural networks [52]: The paper utilized CNN for an image classification task one of the most popular recognition contests [37]. ImageNet contains one of the largest visual databases (ten million URLs of images) as of today. The authors classified 1.2 million images into 1000 different classes and achieved top-1 and top-5 error rates of 37.5% and 17.0%.

Important discoveries from this paper include: Introduction of dropout layer as a means for regularization in a neural network. Since its introduction dropout is widely used in the CNN architecture and has been used in this thesis. Furthermore, the authors trained their network

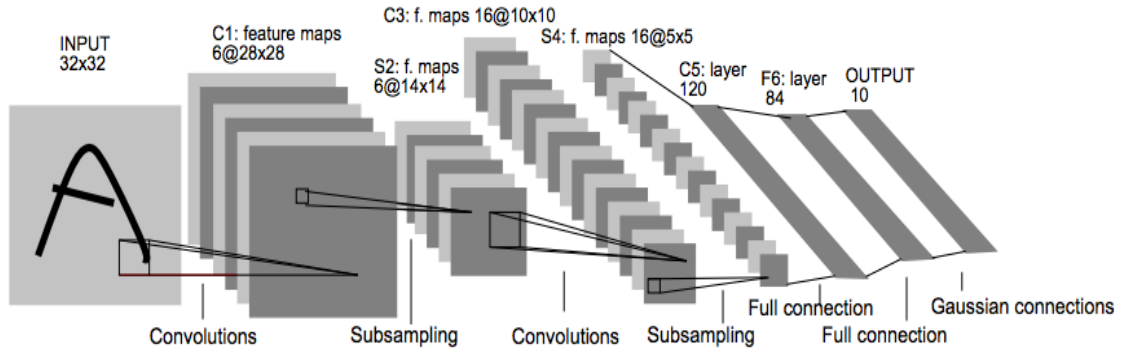


Figure 2.8: Architecture of CNN used in [61]

with cross-GPU parallelization. As of today cross-GPU parallelization is almost a standard approach when handling Big Data. The architecture for the CNN is illustrated in Figure 2.9

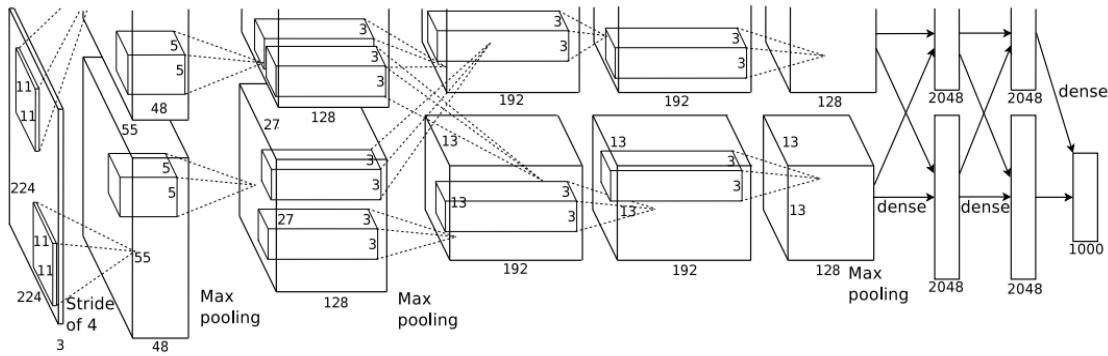


Figure 2.9: Architecture of CNN used in [52]

Very deep convolutional networks for large-scale image recognition [84]: In this paper the authors investigate the effect of the CNN depth on its accuracy in the large-scale image recognition setting. They use an architecture with very small (3x3) convolution filters, which illustrated a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16-19 weight layers. This discovery helped in the decision of choosing 3x3 sized kernel size for the convolutional layer in this thesis. With this approach they secured the first and the second places in the localization and classification tracks respectively in the ImageNet Challenge 2014 submission.

As of today, CNN are under active development by major research organization for different applications. A few important recent developments are the inception v-4 architecture by Google [94] and Deep Residual Networks introduced by Microsoft research [28].

Advantages

In a CNN, convolution layers play the role of feature extractor. Convolutional layers are able to extract the local features because they restrict the receptive fields of the hidden layers to be local. This allows the network to first create good representations of small parts of the input, then assemble representations of larger areas from them [53]. The concepts of sparse interactions, equivariant representations and parameter sharing as explained in the introduction help to improve the accuracy of the model. Trying to get a similar result from MLP would require a larger amount of training because the number of parameters will be much higher as CNN [80], this will lead to a larger effort to regularize the network as the network will be more prone to overfit/underfit.

Challenges

CNN face the same challenges of underfitting and overfitting as mentioned in the MLP challenge background. This is further discussed in the discussion chapter. CNN also require a high computational cost as they are slower to train when compared to MLP. For reference one MLP test for 8/16/8 neurons per layer architecture for 45 min window architecture took ≈ 15 min while the convolutional layer took ≈ 1 hr when trained on a 16 GB memory and 2.7 GHz Intel Core i7 processor computer. This can be an issue when training on more data in the future. However, further systematic testing of various architectures for classification on larger datasets can be done quickly by using Graphics Processing Units (GPUs) [8]. This will significantly lower the training time.

3 Concepts

3.1 Dataset

As energy disaggregation is a relatively new research field, there is a scarcity of publicly available data. An open source data set was chosen for this thesis as this would allow for a comparison of the obtained results with the previous findings. Almost all of the data sets have been made public from 2010 onwards. Among the more popular data sets are Reference Energy Disaggregation Dataset (REDD) [51], UK-Dale [43] and GREEND [68]. As new data sets are being shared it is getting harder to track all of them. This collaborative page helps to share information which is regularly updated [99].

3.1.1 Overview

The data set used for this thesis is the REDD data set and is selected for investigation as the intended purpose for it is to develop disaggregation methods, which can predict, from only the whole-home signal, which devices are being used. It is the most popular data set used currently. The data contains power consumption from real homes, for the whole house as well as for each individual circuit in the house.

The structure of the obtained data set is illustrated in Figure 3.1 and the overview of the data set is illustrated in Table 3.1.

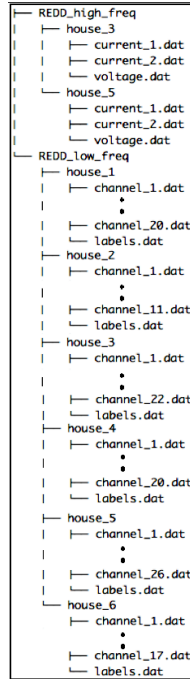


Figure 3.1: REDD data set structure

The "REDD_high_freq" folder contains the voltage and current readings of the appliances for 2 houses at a higher frequency of 15 kHz ($\approx 66 \mu\text{sec}$). The "REDD_low_freq" folder contains the power readings of the appliances for all houses at a sampling rate of 3 sec. "REDD_low_freq" is selected for training and testing as it contains more house and appliance data.

The sub folders of "REDD_low_freq" represent the houses that the data was collected from. Each house contains all the channels (appliances) measured for collecting data. The "label.dat" file in each "House_x" folder maps the names of the "channel.dat" files with their associated appliance name.

In this thesis only the power readings of the appliances are used due to the inconsistencies in the raw data where there are multiple gaps in the time stamps. These inconsistencies are addressed in the next section (**Analysis**). Another reason for not using time readings for the model is that although time of day can help to indicate which appliance is being used, any appliance can be used at non typical times for non typical durations.

Figure 3.2 illustrates the data of the power consumption of a dishwasher from House 1.

House	Appliance Circuits*	Samples	Appliances**
1	18	745,878	oven, refrigerator, dishwasher, kitchen outlets, lighting, washer dryer, microwave, bathroom gfi, electric heat, stove
2	9	318,759	kitchen outlets, lighting, stove, microwave, refrigerator, dishwasher, disposal
3	20	404,107	outlets unknown, lighting, electronics, refrigerator, disposal, dishwasher, furnace, washer dryer, microwave, smoke alarms, bathroom gfi, kitchen outlets
4	18	570,363	lighting, furnace, kitchen outlets, outlets unknown, washer dryer, stove, air conditioning, miscellaneous, smoke alarms, kitchen outlets, dishwasher, bathroom gfi
5	24	80,417	microwave, lighting, outlets unknown, furnace, washer dryer, subpanel, electric heat, bathroom gfi, refrigerator, dishwasher, disposal, electronics, kitchen outlets, outdoor outlets
6	15	376,968	kitchen outlets, washer dryer, stove, electronics, bathroom gfi, refrigerator, dishwasher, outlets unknown, electric heat, lighting, air conditioning

* 2 circuits excluded for each house which measure mains supply (not appliances)

** Note that multiple appliance circuit measurements are taken for the same type of appliance (e.g 3 lighting appliance are measured in house 1)

Table 3.1: REDD data description

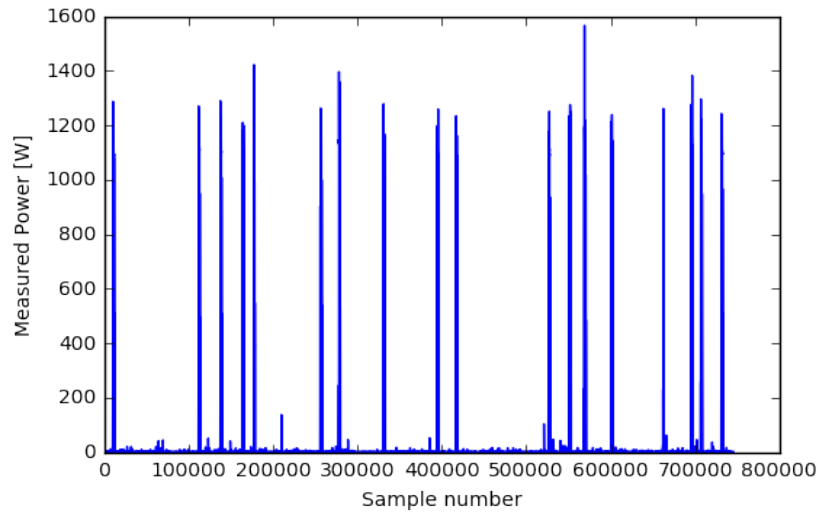


Figure 3.2: Power consumption of dishwasher from house 1

Figure 3.3 illustrates one cycle of dishwasher consumption from house 1. The period stretches from 10,000 to 12,000 samples which corresponds to ≈ 1.67 hours.

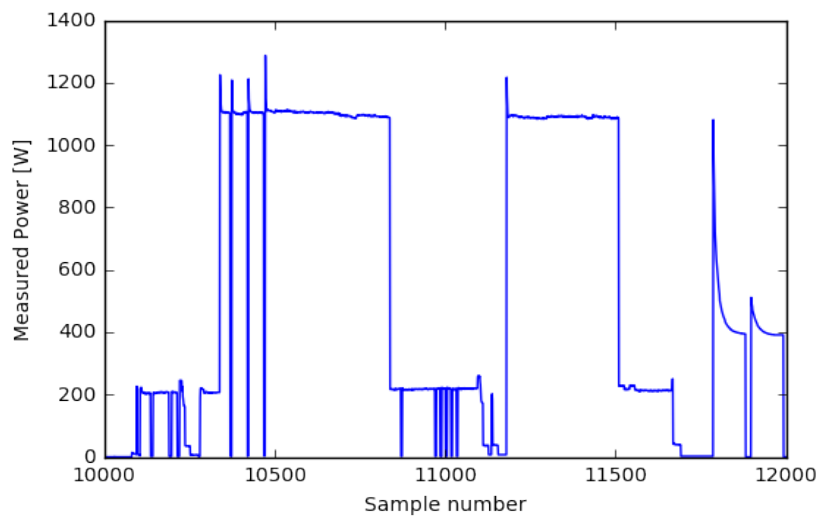


Figure 3.3: A complete cycle of dishwasher consumption from house 1

3.1.2 Analysis

Although the data set is claimed to be sampled every 3 sec for "REDD_low_freq", not all the readings are sampled at equal intervals. As this has the potential to effect the accuracy of the model, an analysis is done on the collected data for each house. Another study comparing data sets found these gaps for the 3 houses that were investigated [5].

House number	Mean sample interval time [sec]	Standard deviation of sample interval time
1	4.2	272.7
2	9.4	3168.3
3	9.6	2649.4
4	7.3	2307.6
5	47.1	9769.1
6	5.3	914.7

Table 3.2: Analysis of sampling period of 3s in "REDD_low_freq" data

Table 3.2 in Appendix-A includes the number of samples that are taken at greater than 3s interval. These values are calculated for each appliance. Using these values the following Table 3.2 was obtained.

Figure 3.4 illustrates the number of samples measured at greater intervals than 3 s in the 6 houses for a dishwasher. Appendix-A contains the figures for lighting in 8.1 and washer dryer in 8.2 respectively. From these figures, it is illustrated that house 6 has the most accurate sampling of data as it has the lowest number of measurement samples relative to the total number of samples across the remaining houses.

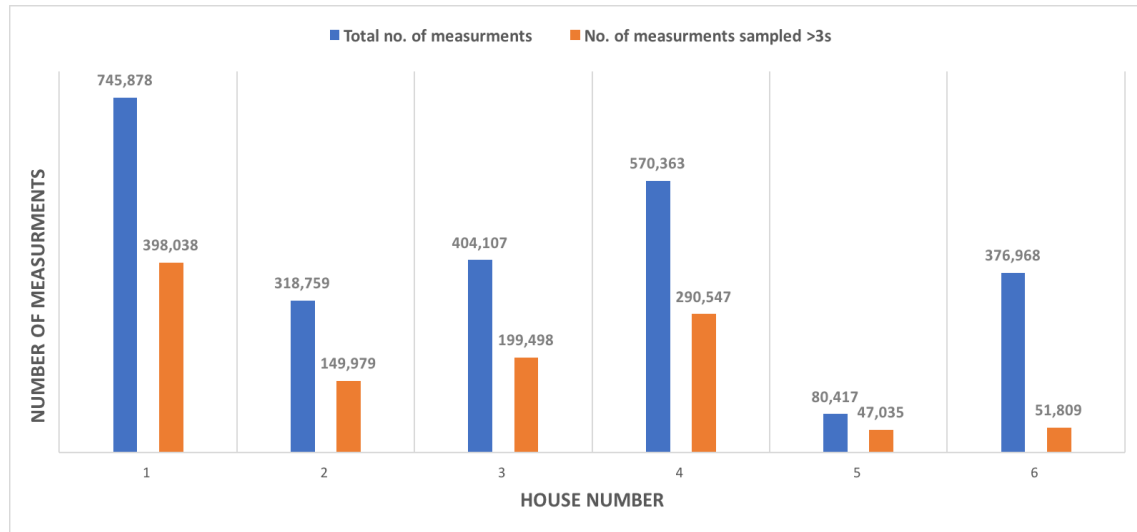


Figure 3.4: Measurement samples from dishwasher

There are evident inconsistencies in the data with the largest being for house 5, where there are samples recorded after 6 days instead of the periodic 3 s. This inconsistency occurs at UTC time stamp of 1306977625 where the next recorded sample is at 1307512396 corresponding to the 6 day delay. Furthermore, the data for all the 6 houses is not recorded at equal 3 s intervals. This is illustrated in Table 8.2 where the number of recorded samples is given for each house with the corresponding number of readings that are sampled at an interval greater than 3 s. Note that a house in the REDD data set can have multiple appliances of the same type. For instance, House 1 has 3 measurements for lighting and 3 washer dryers.

There is no discussion about this missing data from the authors. Nevertheless, a number of reasons can cause the large standard deviation and higher mean value recordings. For instance, in Figure 3.5 the hardware and software setup for the collection of the data is provided. A fault in the recording device or in sending the data from the router to the server could result in missing or faulty data.

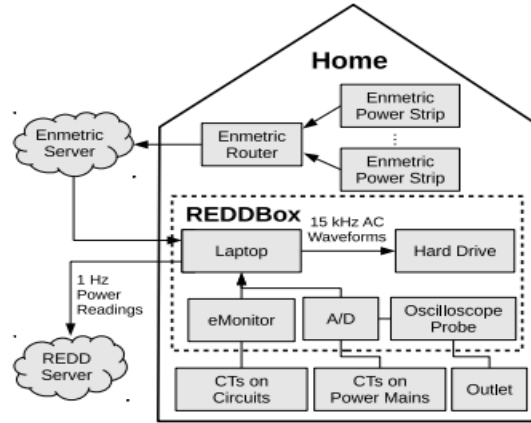


Figure 3.5: Hardware and software setup for REDD dataset [51]

The authors of REDD use commercial devices for recording the data developed by Enmetric. However, they do not specify which device was used for monitoring the power consumption of which appliance. It is important to declare the setup as certain devices are not suitable for measuring certain appliances. As an example, a Power Port device from Enmetric [93] can handle a maximum power load of 1800 W. Appliances in the REDD data set such as refrigerators, lighting etc consume more power than this device can handle. In such cases the data collected could be affected.

3.1.3 Windowing Data

This method of windowing data [15] allows the approach to become a classical supervised learning problem. For neural networks, there are two main ways of incorporating context into sequence processing tasks: collect the inputs into overlapping time-windows, and treat the task as spatial; or use recurrent connections to model the flow of time directly [26]. In this thesis recurrent networks are not implemented due to the limited time, but are promising networks for future work. A discussion for the potential in using these networks is provided in the conclusion chapter. Hence, by windowing one constructs a window classifier that maps an input window of length x into an individual output value y . The window sizes investigated in this thesis are 45 min, 2 hours, 1 day and 3 days as illustrated in Figure 3.6 for a lighting appliance from House 1. These values are empirically derived as window sizes smaller than 45 min have too less data for each x input (<900 samples for each input). This would lead to a case where the behaviour of the power consumption of an appliance will no longer be learned as there are too few samples and lead to a rule based system [96]. The disadvantage of a rule based system is that when introducing new knowledge to solve some

specific problem (for example adding a new rule), one might introduce contradictions with the previous rules [13]. Another reason for choosing the window sizes is, that the time windows when represented as samples, are perfect squares and allow for equal dimension 2D matrix representation $N \times N$ of the input data for the convolutional neural network. Hence, making it easier to choose the filter size, kernel etc. As an example 45 min window corresponds to 900 samples (as each sample is 3 sec) and can be represented as a 30×30 input vector for the convolutional layer. Hence, the raw appliance data is split into the 5 window intervals as part of the data pre-processing step. These values correspond to the sample sizes of the split data which are: 900 (45 min), 2500 (2 hour), 9604 (8 hours), 28900 (1 day) and 86436 (3 days) respectively. Furthermore, the interval from 45 min - 3 days offers a realistic implementation of the approach for use cases which involve faster classification. This is further addressed in the **conclusion** chapter.

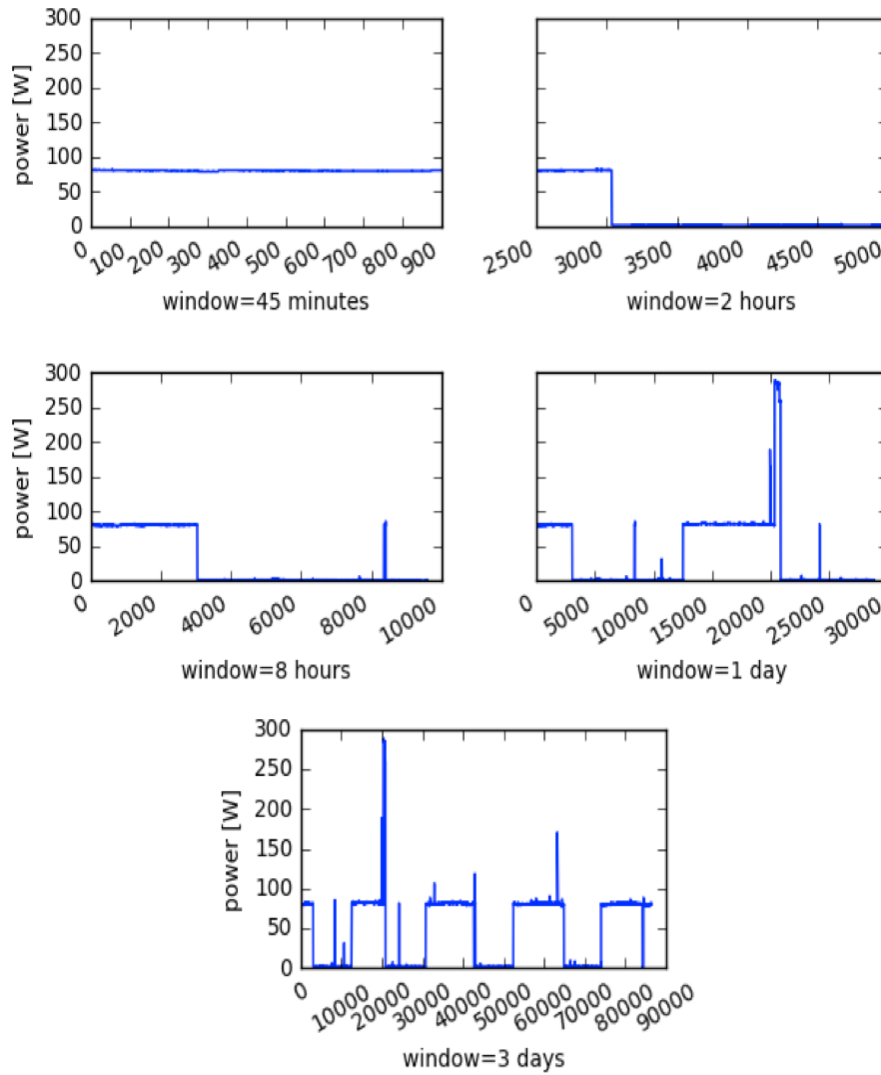


Figure 3.6: Window size visualization for lighting appliance from House 1

3.1.4 Data preprocessing

"REDD_low_freq" is used for the analysis in this thesis. An overview of "REDD_low_freq" can be found in Figure 3.1. This subset was selected as "REDD_low_freq" only contains the data of power mains for 2 houses (House 3 and 5), and therefore does not have appliance level data. Hence, using "REDD_low_freq" enables access to 6 houses for training and testing. Furthermore, using data sampled at a lower frequency illustrates the generalization ability of using deep neural networks.

The raw data from "REDD_low_freq" is illustrated in Figure 3.7.

```

1303171952 8.00
1303171955 8.00
1303171959 8.00
1303171963 8.00
1303171967 3.00
1303171970 219.00
1303171974 219.00
1303171978 221.00
1303171982 223.00
1303171985 213.00
1303171989 213.00
1303171998 207.00
1303172002 208.00
1303172005 205.00
1303172009 204.00
1303172012 206.00
1303172016 206.00
1303172019 205.00
1303172023 205.00
1303172026 205.00
1303172030 202.00
1303172033 205.00
1303172036 201.00

```

Figure 3.7: Raw data from dishwasher from House 1

Each appliance file contains a UTC timestamps (as integers) and power readings which are the recorded apparent power of the circuit. For training the neural network the power readings are used as an input into a dense layer for MLP approach and convolutional layer for the convolutional approach. The time readings are not used because of the missing time stamps as discussed in the Dataset subsection 3.1. The following algorithm illustrates how the data was processed to make new files which only contained power reading values Algorithm:1.

Algorithm 1: Pre-Processing REDD raw data

```

for House 1 to 6 do
    for Appliances 1 to 3 do
        Open channel_x.dat file;
        Select power reading column;
        Save power reading column to new_channel_x.dat file;
        Close channel_x.dat file;
    end
end

```

Implementation Environment

Python is used for processing the raw data. A number of reasons led to using python for processing the data. It is popular among data scientists as many popular machine learning

libraries are written in Python e.g (Keras, Tensorflow, sckit-learn, Theano etc). It provides easy data visualization and analysis framework. It is easy to setup (if you have a Linux distribution it is already installed).

The libraries relevant for processing the data are Pandas [66] which is a data structure and analyses tool, Matplotlib [36] which is used as a plotting library and Numpy [89] which provides powerful N-dimensional array objects to perform linear algebra.

The framework used in this thesis is Keras. Keras is an open source neural network library written in Python. It is capable of running on top of Deeplearning4j, Tensorflow, CNTK or Theano. It is designed to enable fast experimentation with deep neural networks [87]. Keras was chosen as the framework in this thesis as it enables for faster implementation of models and due to its higher abstraction better human readability.

In Keras a sequential model is used to create the architecture for MLP and CNN neural networks. A sequential model is a linear stack of layers [49]. This model needs to know what input shape it should expect. For this reason, the first layer in a sequential model needs to receive information about its input shape. For MLP and CNN architectures the input data into a Keras sequential model is a list of Numpy arrays. The difference is that input for MLP model is 1D whereas input for a 2D convolutional layer is 2D. Keras also includes a 1D and 3D convolutional layer [47]. 1D convolutional layers have been used for audio signals and 2D convolutional layers have been used for image processing and speech processing [59], while 3D convolutional layers have been used for real time object recognition [39].

The windowed approach splits the consumption profile of an appliance in to smaller parts as input to the neural network. Splitting the data was achieved by creating a method that took a defined size of samples for each file. The code for this method has been provided in the CD as explained in the Appendix.

3.2 Appliances Detected

The appliances chosen for the training of the NN models are lighting, dishwasher, washer dryer and unknown appliances. These appliances are chosen as they are the only common appliances among all 6 houses in the REDD dataset. This allows for the neural network to have more data to train and predict on, and thus increases the reliability of the results. This also allows for a prediction of the mentioned 3 common appliances out of unknown appliance. Hence, allowing for a more realistic portrayal of a real life scenario. An overview of this data is provided in Table 3.5 and a list of all appliances is provided in the appendix.

The unknown outlets are readings of appliances in the houses for which there is no label. The signal characteristic differs for each house reading. This is illustrated in Figure 3.8 where the signal of 4 channels is illustrated from the 9 total channels (3.5).

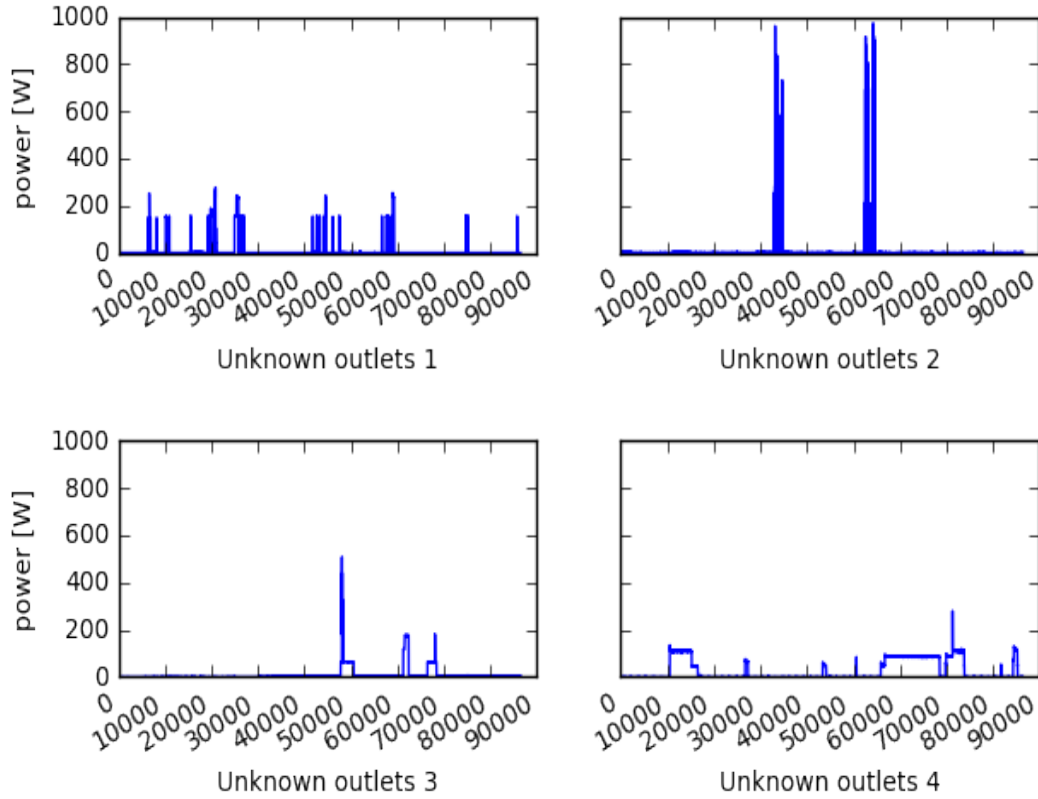


Figure 3.8: Outlets unknown appliance readings for 3 day window size

3.3 Test Structure

The classification accuracy of a model depends on the choice of model and its architecture. The **depth** of the neural network and well as the **number of neurons** play a critical role in the learning of the models. Hence these parameters are investigated in this thesis. It is difficult to select the optimum number of layer for a nueral network. There are methods such as Akaike's Information Criterion to find the number of hidden layers in the NN architecture [69] however it is specific to the data being used and could result in the model not training [74]. Furthermore, the REDD dataset consists of a smaller amount of data when compared to traditional datasets, hence requiring fewer amounts of neurons and layers to train the neural networks.

The data consists of all the samples for each common appliance in all houses as illustrated in the Table 3.3. In order to test the performance of the neural network this data is split into training and testing sets. The test set is 10% of the respective windowed data as illustrates in Table 3.4. Note that data cannot be split in the exact specified window size as there are extra values. In order to keep the specified window size these extra values are discarded. This becomes more evident as the window size increases, a larger proportion of extra values is discarded.

A number of appliances of the same class are expected to have similar power consumption. However, a large variance in power consumption characteristics is present in the "REDD_low_freq" dataset. Figure 3.9 illustrates the differing characteristic in the power consumption of 3 lighting and washer dryer appliances. Note that as there is only one dishwasher appliance reading per house in Table 3.3. In Figure 3.9 the variation of peak power consumption readings for lighting is 1200 W, 50 W and 400 W respectively. While, the variation of peak power consumption readings for washer dryer is 4900 W, 20 W and 3200 W respectively. A typical dishwasher consumes 2790W of power [72]. Therefore, a variation of 1000 W is significant as the power consumption is no longer typical.

Hence, initial tests aimed to classify appliances without the inclusion of unknown appliances in the training data in order to see how well the model can differentiate known appliances. Consequently, unknown appliance data is added to the training data set and tested. This approach allows for a realistic portrayal of classification.

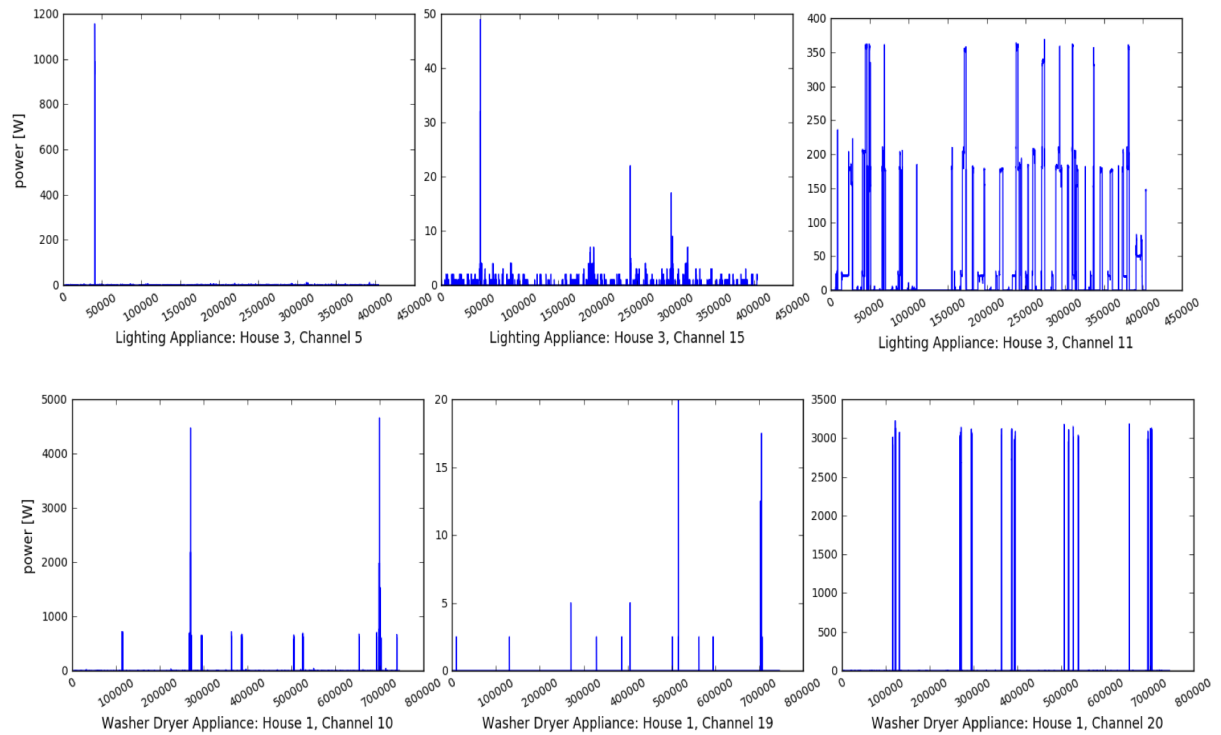


Figure 3.9: Differing Power Consumption of Lighting and Washer Dryer appliances

Table 3.3 provides an overview of the common appliance data for the initial test which do not include unknown appliances. Table 3.4 provides an overview of the corresponding train/testing data generated from the total number of samples obtained from Table 3.3.

Common Appliances in 6 Houses	Total no. of Appliances	Total no. of Samples
Dishwasher	6	2,496,492
Washer Dryer	10	4,472,772
Lighting	19	7,067,070
Total	35	14,036,334

Table 3.3: Overview of common Appliance Data

Window size	Training samples	Testing samples
45 minutes	12,631,140	1,403,460
2 hours	12,629,250	1,403,250
8 hours	12,611,012	1,401,222
1 day	12,588,840	1,398,760
3 days	12,446,784	1,382,976

Table 3.4: Overview of Training and Testing Data for common Appliances

Table 3.5 provides an overview of the common appliance data with the inclusion of unknown appliances. Table 3.6 provides an overview of the corresponding training/testing data generated from the total number of samples obtained from Table 3.5.

Common Appliances in 6 Houses	Total no. of Appliances	Total no. of Samples
Dishwasher	6	2,496,492
Washer Dryer	10	4,472,772
Lighting	19	7,067,070
Unknown Outlets	9	2,777,871
Total	46	16,814,205

Table 3.5: Overview of common Appliance Data with unknown appliances

Window size	Training samples	Testing samples
45 minutes	15,130,800	1,681,200
2 hours	15,129,000	1,681,000
8 hours	15,109,012	1,678,779
1 day	15,085,800	1,676,200
3 days	14,936,140	1,659,571

Table 3.6: Overview of Training and Testing Data for common and unknown appliances

The data in each set is independent from each other, and is identically distributed, drawn from the same probability distribution as each other. This is achieved using the method of splitting the original data into a training and test data set provided in the scikit-learn [81] library.

3.4 Design Flow

This thesis will investigate the different types of Deep Learning neural networks which can be used for modeling data and will determine two which fit best. The first is the Multi-Layer Perceptron (MLP) and the second is Convolutional Neural Network (CNN). MLP provides a baseline approach as it is the simplest of deep neural networks. CNN aims to increase the accuracy of the baseline approach. The architectures, activation functions and optimizers of both neural networks are investigated with the aim of improving accurate modeling and providing optimal configuration for classification. The accuracy is measured using two loss functions- Mean Squared Error and Categorical Cross-Entropy. The design flow of the thesis is illustrated in Figure 3.10. It illustrates the process of classifying and predicting the appliances from the early stage of acquiring the data.

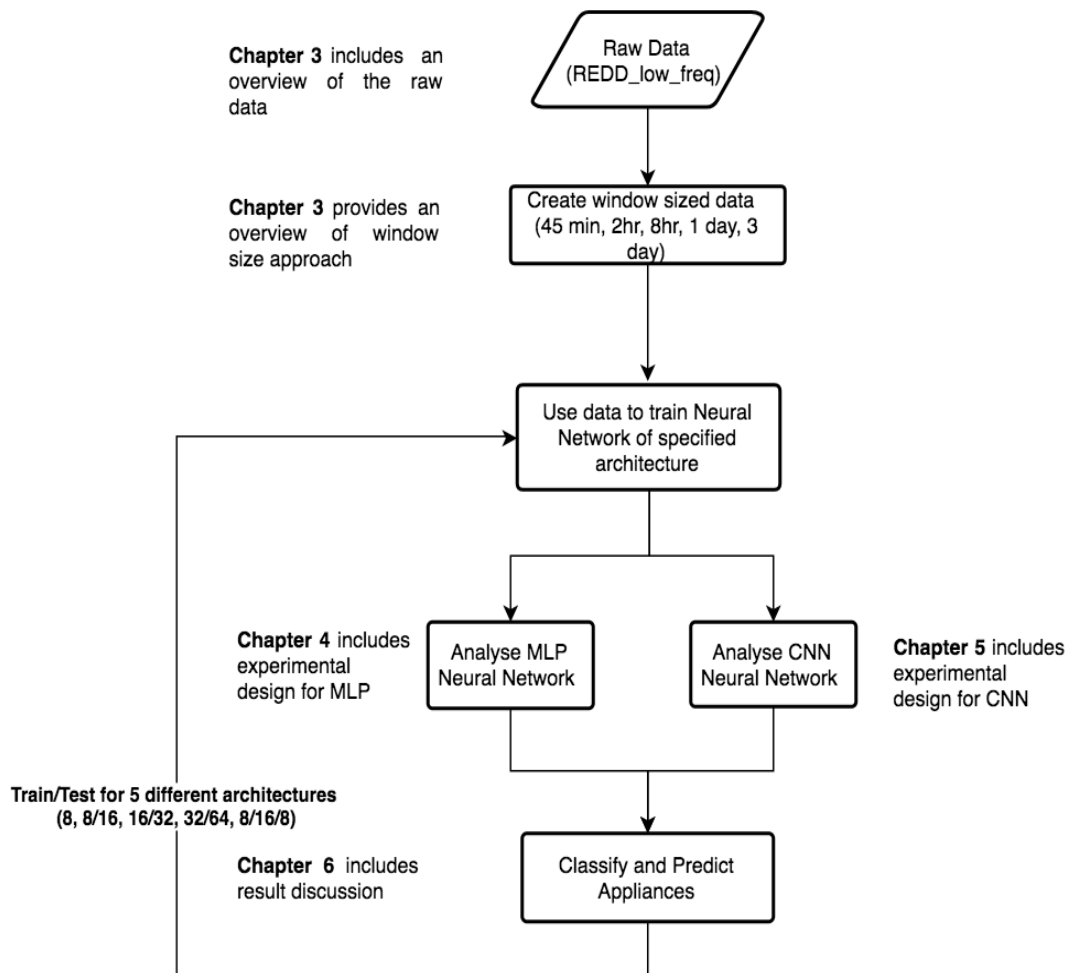


Figure 3.10: Overview of design flow

4 Implementation of Multi-Layer Perceptron

The following chapter discusses the input, parameters and architecture required for implementation of MLP model.

4.1 Implementation Design

In the example provided in Figure 4.1 a representation of the flow of data into the model is visualized. Note that the illustration is for known appliances tests case, for the unknown appliances there will simply be another column added with the depiction of the windowed signal. Additional appliances will also be added in a similar fashion. The windowed data from Dishwasher, Lighting and Washer Dryer is fed into the MLP neural network model for the baseline approach referring to Table 3.3 and Table 3.4. The figure demonstrates the design flow for the configuration of a 32/64 neurons per layer architecture. There are 4 more architectures investigated. Namely, 8 neurons per layer, 8/16 neurons per layer, 16/32 neurons per layer and 8/16/8 neurons per layer. The data windowed data of size n which can be 45 min, 2 hours, 8 hours, 1 day or 3 days, is inputted into an Numpy array. In the simplest case (when size of window = no. of samples and there is only one reading per appliance) This array is then an array of 3 arrays corresponding to dishwasher, lightning and washer dryer. This array of arrays is then fed into the sequential model as depicted in the model section of the illustration.

Tests are then run for different window sizes (45 min, 2 hr, 8 hr, 1 day and 3 days) while using one architecture, in the case of the illustration 32/64 neurons per layer. Dropout is used as a method for regularization. The corresponding fraction (0.2) represents the input units to drop. For each layer their is an activation function which is softsign for the hidden layers. The activation function was empirically tested out of the available activation function in the Keras library as it reduced the error function most compared to the others. In the last layer (output layer) a softmax activation function is used. This was chosen as it will output a separate probability for each of the appliance classes, and the probabilities will all add up to 1 [95].

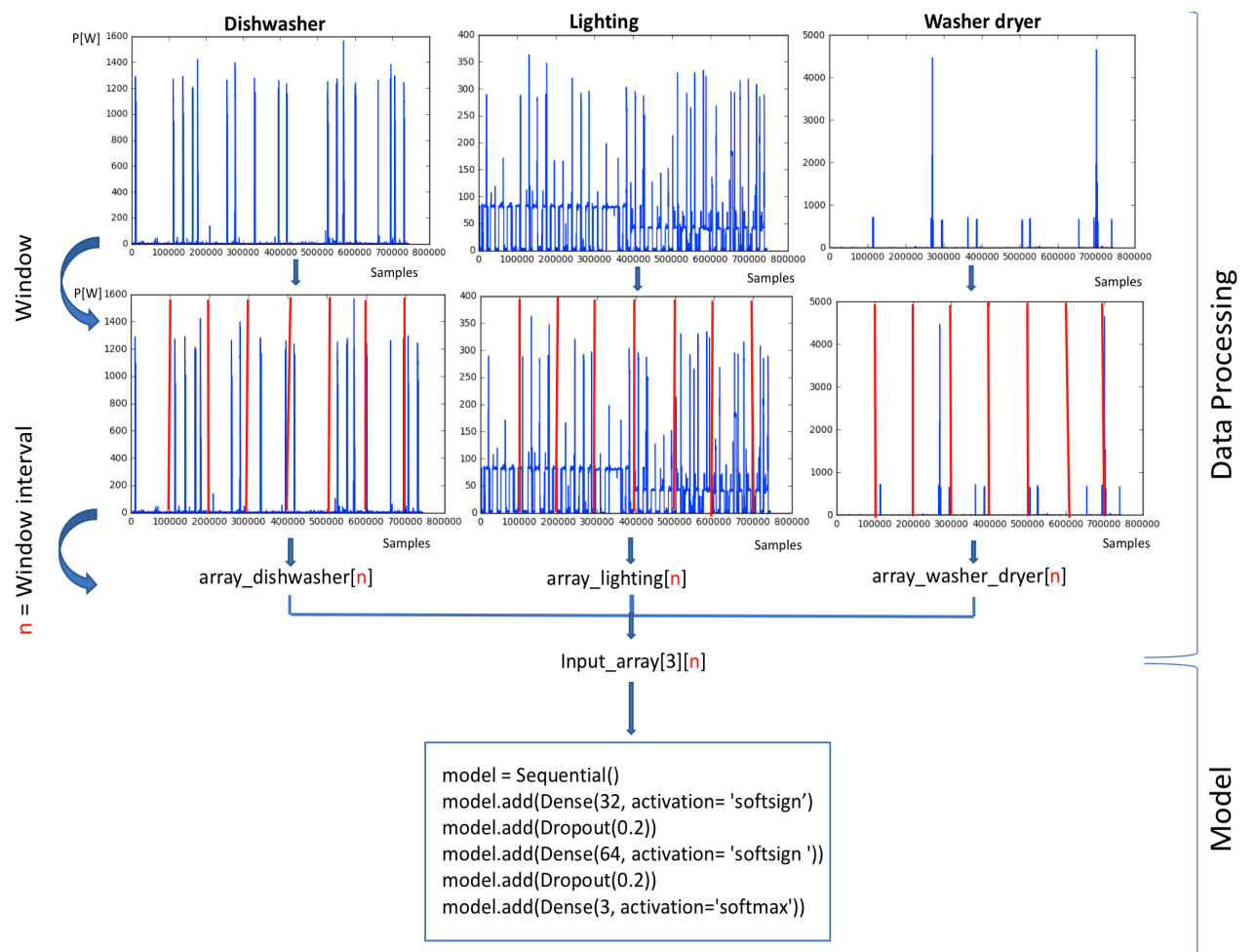


Figure 4.1: Basic design architecture for MLP

Two loss functions (MSE and CE) are used to compare the validation accuracy for better reliability of the obtained result. The model as implemented in Keras is provided in the attached code.

4.2 Results

Neurons per Layer	45 min V.A [%]		2 hr V.A [%]		8 hr V.A [%]		1 day V.A [%]		3 days V.A [%]	
	MSE	CE	MSE	CE	MSE	CE	MSE	CE	MSE	CE
8	75.11	75.66	77.22	77.58	76.48	76.03	73.47	70.07	70.83	64.58
8/16	76.15	75.90	78.71	76.75	73.97	70.32	67.35	71.43	66.67	68.75
16/32	76.11	76.39	77.46	78.83	73.97	73.74	71.43	70.75	68.75	60.42
32/64	76.43	76.03	77.64	76.16	74.89	71.00	70.75	71.43	68.75	68.75
8/16/8	76.05	75.38	77.94	76.99	78.54	80.37	75.51	75.51	68.75	68.75

Table 4.1: The effect of window size and loss function on validation accuracy for MLP architecture

The effect window size has on the validation accuracy is quite evident across the different architectures. The accuracy peaks at the 2 hr window and drops for the following window sizes. It is important to note that the model architecture across the neurons per layer remains the same for the different window sizes. Hence the difference is the amount of data the model takes for each input array. A 3 day time window will result in the input having 86,436 samples, while a 45 window will have 900 samples for each input. Hence, in such a case 8 neurons are not sufficient to model the amount of data as illustrated by the reduced validation accuracy.

The validation accuracy when MSE and CE error are used as loss functions has the smallest difference for 45 min and 2 hr time windows. The largest difference in the validation accuracy between MSE and CE for 45 min window size is 0.67%, 2 hr is 1.96%, 8 hr is 3.89%, 1 day is 4.08% and 3 day is 8.33%. Hence, both loss functions have a closer validation accuracy for smaller window size, and higher discrepancy for larger window size. Thus, smaller window size provide a more reliable result.

The higher validation accuracies are represented by the 8/16/8 neurons per layer architecture. The highest accuracy achieved is 80.37% for 8/16/8 neurons per layer and 8 hr window size, while the lowest accuracy is 60.42% for 16/32 neurons per layer and 3 days window size. Furthermore, the highest accuracy for each neurons per layer configuration have been

made bold in Table 4.1, which illustrate a trend of increasing accuracy with the depth of the neural network.

Neurons per Layer	45 min V.A [%]		2 hr V.A [%]		8 hr V.A [%]		1 day V.A [%]		3 days V.A [%]	
	MSE	CE	MSE	CE	MSE	CE	MSE	CE	MSE	CE
8	75.48	75.30	76.37	76.03	72.95	76.76	72.41	77.01	65.00	60.00
8/16	76.46	75.37	76.28	69.84	64.19	72.38	74.71	70.11	60.00	58.33
16/32	75.50	74.80	75.78	75.88	75.05	70.29	72.99	67.24	61.67	58.33
32/64	76.55	74.88	75.98	75.19	76.38	69.14	74.14	75.29	60.00	56.67
8/16/8	76.28	74.82	76.13	76.37	77.14	74.48	71.84	75.86	68.33	65.00

Table 4.2: The effect of window size and loss function on validation accuracy for MLP architecture with unknown appliances

With the inclusion of unknown appliance data the higher accuracies are obtained for window sizes smaller than 1 day.

The validation accuracy when MSE and CE error are used as loss functions has the smallest difference for 45 min and 2 hr time windows. The largest difference in the validation accuracy between MSE and CE for 45 min window size is 1.46%, 2 hr is 6.44%, 8 hr is 8.19%, 1 day is 5.75% and 3 day is 5.00%. Hence, both loss functions have a closer validation accuracy for smallest window size, and higher discrepancy for larger window sizes. Thus, smaller window size provides a more reliable result.

The highest accuracy achieved is 77.14% for 8/16/8 neurons per layer and 8 hr window size, while the lowest accuracy is 60.00% for 8 neurons per layer and 3 days window size. Furthermore, the highest accuracy for each neurons per layer configuration have been made bold in Table 4.2.

5 Implementation of Convolutional Neural Network

The following chapter discusses the input, parameters and architecture required for implementation of CNN model.

5.1 Implementation Design

In the example provided in Figure 5.1 a representation of the flow of data into the model is visualized. Note that the illustration is for known appliances tests case, for the unknown appliances there will simply be another column added with the depiction of the windowed signal. Additional appliances will also be added in a similar fashion. The windowed data from Dishwasher, Lighting and Washer Dryer is fed into the CNN neural network model for the baseline approach referring to Table 3.5 and Table 3.6. The figure demonstrates the design flow for the configuration of a 32/64 filters per layer architecture. There are 4 more architectures investigated. Namely, 8 filters per layer, 8/16 filters per layer, 16/32 filters per layer and 8/16/8 filters per layer. The data windowed data of size n which can be 45 min, 2 hours, 8 hours, 1 day or 3 days, is inputted into an Numpy array. In the simplest case (when size of window = no. of samples and there is only one reading per appliance) This array is then an array of 3 arrays corresponding to dishwasher, lightning and washer dryer. This array of arrays is then fed into the sequential model as depicted in the model section of the illustration.

Tests are then run for different window sizes (45 min, 2hr, 8 hr, 1 day and 3 days) while using one architecture, in the case of the illustration 32/64 filters per layer. In the convolutional layer the filters are the number of neurons[47] and are illustrated as 32 and 64 for the 2 layers. Kernel size as explained in the introduction is 3X3. Using a small kernel size captures fine details and is regularly used for convolutional models [77][52]. The pool size is 2X2 and is max pooling is performed after each convolutional layer. Dropout is used as a method for regularization. The corresponding fraction (0.2) represents the input units to drop. For each layer their is an activation function which is softsign for the hidden layers. The activation function was empirically tested out of the available activation function in the Keras library as

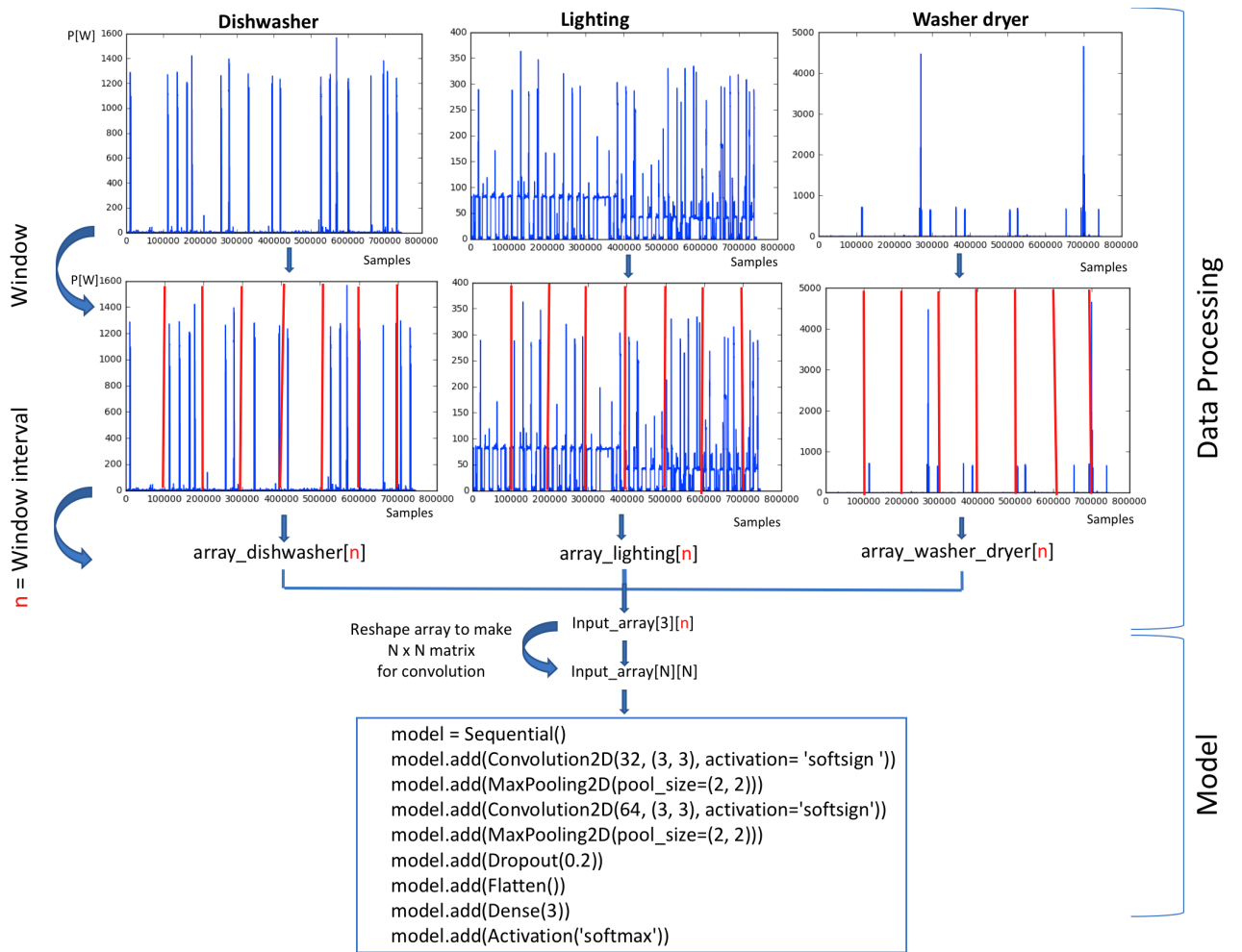


Figure 5.1: Basic design architecture for CNN

it reduced the error function most compared to the others. In the last layer (output layer) a softmax activation function is used. This was chosen as it will output a separate probability for each of the appliance classes, and the probabilities will all add up to 1 [95].

Two loss functions (MSE and CE) are used to compare the validation accuracy for better reliability of the obtained result. The model as implemented in Keras is provided in the attached code.

5.2 Results

Filters per Layer	45 min V.A [%]		2 hr V.A [%]		8 hr V.A [%]		1 day V.A [%]		3 days V.A [%]	
	MSE	CE	MSE	CE	MSE	CE	MSE	CE	MSE	CE
8	79.40	75.73	78.00	76.63	84.25	82.42	86.39	83.67	83.33	79.17
8/16	80.47	80.06	82.74	81.73	85.16	86.76	85.71	83.67	89.58	91.67
16/32	79.25	80.19	81.26	81.91	83.79	85.84	87.07	84.35	87.50	87.50
32/64	79.89	79.32	78.88	79.36	84.02	84.25	83.67	85.71	87.50	87.50
8/16/8	81.35	81.43	85.41	83.51	89.73	88.81	91.16	91.84	95.83	93.75

Table 5.1: The effect of window size and loss function on validation accuracy for CNN architecture

The effect window size has on the validation accuracy is quite evident across the different architectures. The accuracy peaks at the 3 day window and drops for the smaller window sizes.

The validation accuracy when MSE and CE error are used as loss functions has the smallest difference for 45 min and 2 hr time windows. The difference in the validation accuracy between MSE and CE for 45 min window size is 3.67%, 2hr is 1.37%, 8hr is 2.05%, 1 day is 2.72% and 3 day is 4.16%. Hence, both loss functions have a closer validation accuracy for 2 hr and 8 hr window size, and higher discrepancy for 45min and 3 day window size.

The highest accuracy achieved is 95.83% for 8/16/8 filters per layer and 3 day window size, while the lowest accuracy is 75.73% for 8 filters per layer and 45 min window size. Furthermore, the highest accuracy for each filters per layer configuration have been made bold in Table 5.1. As illustrated the validation accuracies for all the filters per layer architecture increase for larger window sizes.

Filters per Layer	45 min V.A [%]		2 hr V.A [%]		8 hr V.A [%]		1 day V.A [%]		3 days V.A [%]	
	MSE	CE	MSE	CE	MSE	CE	MSE	CE	MSE	CE
8	77.05	76.00	78.01	76.92	79.81	79.62	75.29	78.16	63.33	60.00
8/16	79.28	77.84	80.83	79.69	84.76	83.24	79.31	79.89	63.33	60.00
16/32	78.71	77.94	82.22	80.04	84.38	82.29	78.74	81.03	63.33	60.00
32/64	79.28	79.57	81.33	80.14	83.24	83.24	76.44	77.59	60.00	60.00
8/16/8	78.93	77.77	81.08	78.85	83.62	85.90	90.00	84.48	85.00	76.67

Table 5.2: The effect of window size and loss function on validation accuracy for CNN architecture with unknown appliances

With the inclusion of unknown appliance data the higher accuracies are obtained for 8 hr window sizes.

The validation accuracy when MSE and CE error are used as loss functions has the smallest difference for 45 min, 2 hr and 8 hr time windows. The largest difference in the validation accuracy between MSE and CE for 45 min window size is 1.44%, 2 hr is 2.23%, 8 hr is 2.28%, 1 day is 5.52% and 3 day is 8.33%. Hence, both loss functions have a closer validation accuracy for smaller window size, and higher discrepancy for larger window sizes.

The highest accuracy achieved is 90.00% for 8/16/8 filters per layer and 1 day window size, while the lowest accuracy is 60.00% for 8 filters per layer and 3 days window size. Furthermore, the highest accuracy for each filters per layer configuration have been bolded in Table 5.2.

6 Evaluation

In this thesis MLP and CNN neural networks are used for energy disaggregation. This is currently the first supervised classification approach used across all 6 houses on the REDD dataset to the authors knowledge. The results reveal that both MLP and CNN neural network are able to accurately classify household appliances. CNN network performed much better than MLP network.

Window size impacts the accuracy of a Neural Network

Using time-windows has a major drawback. The optimal window size is task dependent. When it is too small the neural network will neglect important information, and too large it will overfit on the training data [26]. In order to solve this issue 5 window sizes were investigated in this thesis which covered small to large window sizes for the REDD dataset. This method allows for the availability of a wide range of window sizes from which the challenges of overfitting and underfitting can be resolved. This is visually illustrated in the Figure 6.1 where a typical behaviour of an overfitt, underfitt and optimal fit is illustrated.

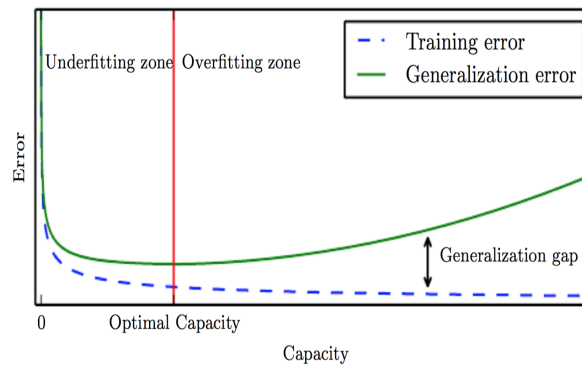


Figure 6.1: Behaviour of overfitt, underfitt and optimal capacity [25]

The behaviour of overfitting is noticed for the 3 days window size for both neural networks for the tested architectures. This behaviour is illustrated when comparing the Figure 6.1 to the obtained Figure 6.2 for 8 neurons per layer architecture and 3 day window size using

MLP. Furthermore, this behaviour is also noticed, albeit less pronounced, in the CNN results Figure 6.3. The gap between the training and test error increased over epochs for both MLP and CNN, while the training error decreased. Hence, this has an effect on the results (validation accuracy) for classification and is also reflected in the respective results tables for MLP Table 4.2 and CNN Table 5.2 where 3 day window size performs worst for both neural networks.

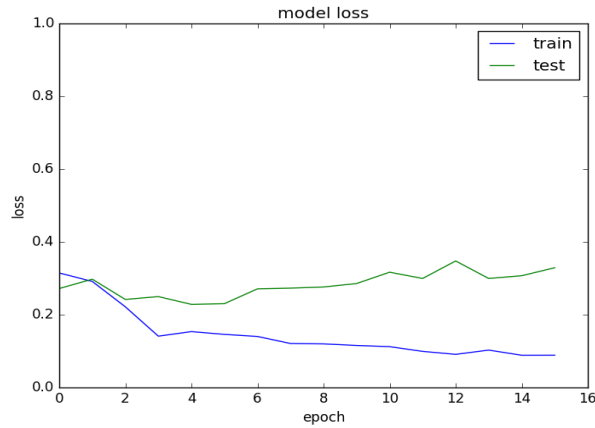


Figure 6.2: Validation loss (error) of MLP over epochs for 3 day window size and 8 neurons per layer architecture

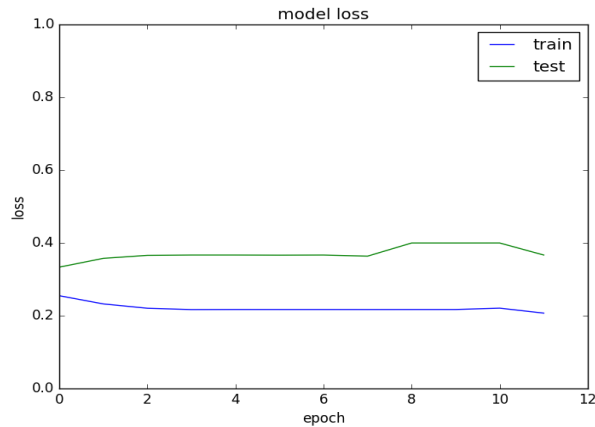


Figure 6.3: Validation loss (error) of CNN over epochs for 3 day window size and 8 neurons per layer architecture

Referencing the results for both the MLP Table 4.2 and CNN Table 5.2 it is illustrated that

MLP performs better for shorter window sizes (45 min and 2 hr) while CNN performs better for larger window sizes (8 hr and 1 day). This is also noticed in the results table for the appliances without unknown appliances in MLP 4.1 and CNN 4.1. The major difference between the two neural networks is how they transform the data as input. When using a 2D convolutional layer the data is reshaped (e.g 900 samples to 30 X 30 samples). This reduced the number of samples per dimension. According to this research findings [34] obtaining a higher level representation (from 1D to 2D) of sentence structure allowed them to capture the rich matching pattern in the structure, thus increasing the accuracy of their model. Nevertheless, for future work 1D convolutional networks are a promising network for investigation.

MLP highest accuracy of **77.14%** was achieved for 8 hr window size while CNN highest accuracy of **90.00%** was achieved for 1 day window size with the inclusion of unknown appliances in the training and testing data. While, in the results obtained without unknown appliances MLP achieved highest accuracy of **80.37%** again for 8 hr window size and CNN achieved highest accuracy of **95.83%** for 3 day window size.

It is feasible to implement learning models for faster detection as the highest accuracies for the 45 min window size for MLP is 76.43% for known appliances and 76.55% with the inclusion unknown appliances, while results for CNN for 45 min window are 81.43% for known appliances and 79.57% with the inclusion of unknown appliances. Do note that this thesis provides a baseline approach for both MLP and CNN networks. The models provided have a lot of margin for parameter tuning which can increase the accuracy further. With a larger amount of data to train on, the accuracy for the classification of appliance is expected to increase. A summary of which is provided in the section "More data improves Neural Network Capability".

Neural Network architecture affects accuracy of prediction

The depth (number of layers) in a neural network has an effect on the accuracy of the models. For both MLP and CNN networks the accuracy for an 8 neurons for one layer architecture resulted in a performance of **77.58%** and **86.39%** respectively as illustrated in Table 6.1. The performance increased significantly when compared to the 8/16/8 neurons per layer architecture for MLP and CNN which resulted in **80.37%** and **95.83%** accuracy respectively. Another interesting finding is that for the 2 layer architecture there is no significant increase in the accuracy when increasing the number of neurons, as 8/16 architecture performs better than 16/32 and 32/64 architecture for both MLP and CNN. Therefore leading to the observation that the number of layers had a greater effect on the accuracy than the number of neurons when investigating the architecture. It is important to note that choosing the number of neurons also depends on the size of the data, which has to be empirically determined for best results. According to this study [18] which investigates the influence of neural network training with respect to architecture depth, model capacity and number of training examples,

increasing depth maximizes the probability of finding poor apparent local minima and thus improving generalization capability of the neural network. In another study [82], the authors show that deeper models were more efficient and effective than shallow models in speech acoustic modeling.

Neurons per Layer	Highest MLP V.A[%]	Highest CNN V.A[%]
8	77.58	86.39
8/16	78.71	91.67
16/32	78.83	87.50
32/64	77.64	87.50
8/16/8	80.37	95.83

Table 6.1: Highest accuracy of each architecture for MLP and CNN networks from Table 4.1 and Table 5.1

Higher performance accuracy from CNN than MLP

This thesis is the only work currently (to the authors knowledge) which has used MLP neural networks for energy disaggregation on the REDD dataset. Therefore, this approach aims to be a baseline for the comparison of other neural networks. The findings of this thesis illustrate that CNN performed significantly better than MLP. This behaviour of CNN achieving a higher accuracy than MLP is also noticed in other works such as on the MNIST dataset. The MNIST dataset is a popular dataset in the machine learning community which has been made for handwritten digit classification [62]. In the best approach available from the Keras examples, MLP[86] networks produce a lower accuracy than CNN[77] networks. Another study concluded that when used in real-time CNN is significantly better than MLP when classifying characters [92].

Comparing the MLP results to the related work, MLP achieves an accuracy of 77.14% compared to the highest achieved overall accuracy of 81.5% [41]. While being a template for Deep Learning approach MLP is also significantly faster at training than CNN. Thus, providing a quick feedback for the researcher/programmer for fine tuning the parameters in the NN model, e.g number of neurons, number of layers, etc. The quick feedback can then be used and applied to CNN architecture as reference.

Compared to MLP, CNN achieved an accuracy of 95.83% on known appliance data and 90.00% with the inclusion of unknown appliance data. The results obtained are more accurate than the highest achieved accuracy of 81.5% in the related work section. This further illustrates how promising neural networks as an approach to energy disaggregation.

More data improves Neural Network Capability

The REDD dataset has a maximum of 12 days of data from only six houses. Comparatively another dataset, mentioned earlier in the dataset section, the UK-Dale dataset has hundreds of days of data available for 5 houses [43]. Thus, proving to be promising for using as a data source.

A Machine/Deep Learning algorithm is able to generalize accurately if it has a variety in the training set. This influenced the author's decision for choosing dishwasher, lighting and washer dryer as the tested appliances since these appliances are present in all 6 houses and there is more data to capture the different power consumption trends for each appliance. Using these three appliances this thesis tests on more data than in the previous approaches where the maximum number of houses used for training and testing is 4 [16][51][50][41].

For neural networks to learn extremely well, training needs to be done on a large amount of data as this will help the networks to generalize well on a wide variety of appliances. According to a study investigating the effect of dataset size on artificial neural network classification [20] states that if training set size are important variable in artificial neural networks and found that the validation accuracy of their artificial neural network increased significantly from 55% to 77%. However, it is important to note that with the increase in the size of the data the architecture of the neural network should change to adapt to the complexity of the newly formed dataset.

Performance on REDD dataset compared to related work

The results of this thesis demonstrate the ability of neural networks to perform well on unknown data. This is an advantage compared to HMM chain approach mentioned in the related work as HMMs do not explicitly encode state duration [42]. Although they are well suited to application domains like speech recognition where the duration of each state does not vary much. Some appliance states might last one minute (e.g. a toaster) whilst other states might last for hours (e.g. washing machine). Furthermore, HMMs typically require that every appliance is modelled. In all the related work the authors don't include unknown appliances in their training data. Hence, their training is done on labelled data for a number of appliances which doesn't include unknown appliances. The table below compares the accuracy of other approaches to this thesis.

Research papers	V.A[%]
J. Z. Kolter and M. J. Johnson [51] with known appliances	47.7%
J. Z. Kolter and T. S. Jaakkola [50] with known appliances	60.3%
M. J. Johnson and A. S. Willsky [41] with known appliances	81.5%
Thesis MLP approach with known appliances	80.37%
Thesis MLP approach with unknown appliances	77.14%
Thesis CNN approach with known appliances	95.83%
Thesis CNN approach with unknown appliances	90.00%

Table 6.2: Comparison of results with related work

7 Conclusion and Future Work

The aim of this thesis was to accurately classify household appliances in a short window interval based on their power consumption data. The results illustrate that neural networks can be used to accurately classify household appliances. This thesis is currently the first supervised classification approach used across all 6 houses on the REDD dataset to the authors knowledge. It was also of significant importance to use an open source dataset (REDD) to allow a comparison of results to the related work of researchers.

Many MLPs and CNNs with different architectures, optimization algorithms, and activation functions were tested in this thesis. A detailed description of the applied preprocessing steps and used architectures enables others to reproduce these results and use this thesis as a base for using deep neural networks for energy disaggregation.

Out of the two neural networks investigated CNN perform much better than MLP. It is also observed that the depth of the neural network increased the accuracy for classification. The highest achieved accuracy from MLP and CNN were for a 3 layer architecture. This thesis is currently the only work which includes unknown data in the training data. Hence, the results for both known and unknown appliances are included to compare to the related work and investigate the ability of the NNs when predicted unseen data. With the inclusion of unknown appliances in the training and testing data, MLP highest accuracy of **77.14%** was achieved for 8 hr window size for 8/16/8 neurons per layer architecture while CNN highest accuracy of **90.00%** was also achieved for 8/16/8 neurons per layer architecture for 1 day window size. Without unknown appliances MLP achieved the highest accuracy of **80.37%** for 8/16/8 neurons per layer architecture again for the 8 hr window size and CNN achieved highest accuracy of **95.83%** for 8/16/8 neurons per layer architecture for the 3 day window size. It is expected that the accuracy of an algorithm decreases when unknown appliances are added in the data pool. Nonetheless, the result of CNN approach of **90.00%** accuracy for known appliances is higher than the top accuracy of **81.5%** from the related work which doesn't include unknown appliances.

Another discovery was that MLP network performed best for window sizes smaller than 8 hr while CNN performed best for window sizes greater than 8 hr. CNN overall performed better than MLP for all window sizes and neural network architectures. Since the conducted experiments produced promising results for CNN, it is viable to focus future research on the improvement of this CNN architecture to achieve higher accuracy.

A limitation for this thesis is that despite the efforts to include as much data over 6 houses, only 3 appliances are classified. Therefore, having access to more appliances which are common among all houses will increase the classification ability of the NN models. Nonetheless, the model architecture for both MLP and CNN as detailed in the thesis allows for the addition of new appliances.

In interest of adding more data to improve the models Lambda architecture can be used for the deployment of Deep Learning models in a production setting. Lambda architecture is an generic architectural pattern which can be used for a data processing architecture to handle streams of data (Big Data) for a Deep Learning model. An overview of this architecture is illustrated in Figure 7.1. The Lambda architecture provides a real time prediction/result from the Deep Learning model. Furthermore, this architecture allows for the NN model to be retrained to reflect the updated parameters over time e.g additional new appliances in the house. The architecture consists of a **batch layer** which aims at achieving perfect accuracy by being able to process all the available data. The best trained models of MLP and CNN from this thesis can be trained in this layer. The **speed layer** processes data streams in real time. The trained model in the batch layer is deployed to the speed layer for real time prediction. The updated model in the batch layer can be redeployed at a regular frequency (e.g every 2 days) to the speed layer. This frequency can be chosen to see if the updated models accuracy is greater than the previous models. If it isn't, then the speed layer continues to use the older model. Finally, there is a **serving layer** which stores the results from the batch and speed layer. These results can be obtained by querying the database. Additional applications can be built for querying to the database and providing the user a visual dashboard of the appliance usage.

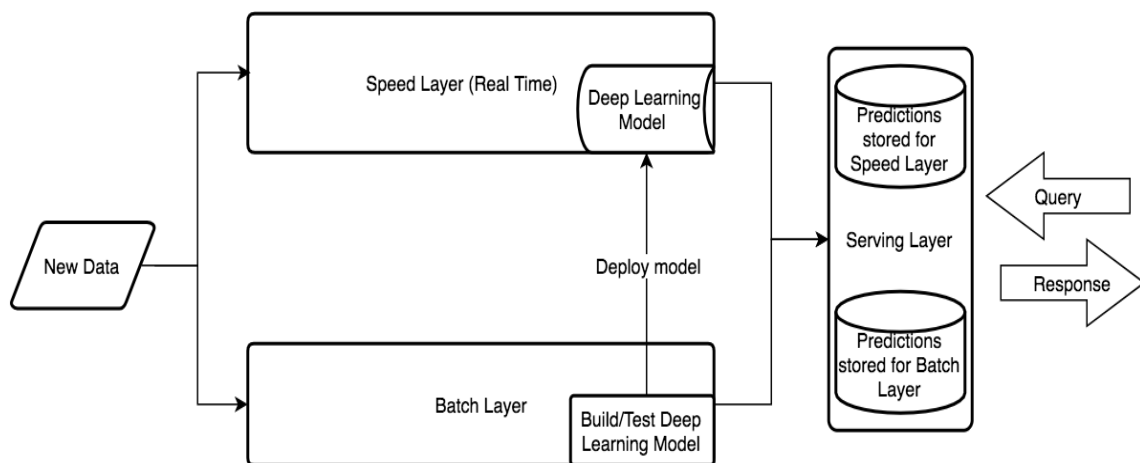


Figure 7.1: Overview of Lambda architecture

The author of this thesis contacted the original authors of the REDD dataset to inquire about

the missing values. However, they did not give an explanation for the missing values which could help to fill the missing values. Hence, these missing values, as identified in the data analysis section, can be filled using statistical methods, last observed value, moving window average [35] etc as a preprocessing step in the future to have a complete dataset. In reference to the preprocessing steps, the data in this thesis was not processed for noise. This can also be dealt at the hardware level, where using proper grounding methods, shielded and twisted wires, signal averaging methods, filters, and differential input voltage amplifiers can control the noise in most measurements [10]. This can help to increase the accuracy of the models. Nevertheless, as the power consumption data is not preprocessed, the obtained results highlight the generalization ability of MLP and CNN.

Since the data used for training the neural network models in this thesis consists only of the power consumption, a possibility for future work is to use a dataset which has complete time stamped data corresponding to the power consumption values. This will add the feature of time to determine if there is a correlation between power consumption of an appliance and the time of the day. If there is, this can help to increase the accuracy of the model. Another possibility is to add the feature of house when training the model. This could further improve the accuracy of the model as a more detailed representation will be available. However, more research will need to be done to determine the existence and extent of any such correlation.

This thesis covers MLP and CNN networks. In the field of Deep Learning other promising networks include Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM). RNNs are distinguished from feedforward networks by a feedback loop, ingesting their own outputs moment after moment as input to the layers. Creating a sort of memory in the network. This can be helpful for this dataset as the past sequence of an appliance can be stored in the memory of network [85] and treat the data as time series as RNNs are extensively used for time series data [11]. LSTM is a variation based on RNNs which can store temporal information using recursive connections. They are designed to overcome the vanishing and exploding gradient problem which are inherent to RNNs [31]. Thus, using them could further increase the accuracy obtained from a simple RNN implementation.

The author hopes that his work will serve as a useful foundation for future research concerning the use of artificial neural networks in energy disaggregation.

8 Appendix A

This appendix contains the analysis of the "REDD_low_freq" folder.

The table below contains the overview of data in the "REDD_low_freq" folder used for this thesis

Channel no.	House 1 appliance	House 2 appliance	House 3 appliance	House 4 appliance	House 5 appliance	House 6 appliance
1	mains	mains	mains	mains	mains	mains
2	mains	mains	mains	mains	mains	mains
3	oven	kitchen_outlets	outlets_unknown	lighting	microwave	kitchen_outlets
4	oven	lighting	outlets_unknown	furnace	lighting	washer_dryer
5	refrigerator	stove	lighting	kitchen_outlets	outlets_unknown	stove
6	dishwasher	microwave	electronics	outlets_unknown	furnace	electronics
7	kitchen_outlets	washer_dryer	refrigerator	washer_dryer	outlets_unknown	bathroom_gfi
8	kitchen_outlets	kitchen_outlets	disposal	stove	washer_dryer	refrigerator
9	lighting	refrigerator	dishwasher	air_conditioning	washer_dryer	dishwasher
10	washer_dryer	dishwater	furnace	air_conditioning	subpanel	outlets_unknown
11	microwave	disposal	lighting	miscellaneous	subpanel	outlets_unknown
12	bathroom_gfi		outlets_unknown	smoke_alarms	electric_heat	electric_heat
13	electric_heat		washer_dryer	lighting	electric_heat	kitchen_outlets
14	stove		washer_dryer	kitchen_outlets	lighting	lighting
15	kitchen_outlets		lighting	dishwasher	outlets_unknown	air_conditioning
16	kitchen_outlets		microwave	bathroom_gfi	bathroom_gfi	air_conditioning
17	lighting		lighting	bathroom_gfi	lighting	air_conditioning
18	lighting		smoke_alarms	lighting	refrigerator	
19	washer_dryer		lighting	lighting	lighting	
20	washer_dryer		bathroom_gfi	air_conditioning	dishwasher	
21			kitchen_outlets		disposal	
22			kitchen_outlets		electronics	
23					lighting	
24					kitchen_outlets	
25					kitchen_outlets	
26					outdoor_outlets	

Table 8.1: Complete data description of REDD_low_freq folder

The table below contains the information used for the analysis of the "REDD_low_freq" dataset.

House no.	Total no. of recorded samples	Dishwasher samples > 3s	Lighting samples > 3s	Washer dryer samples > 3s
1	745878	398038	398041, 397960, 398044	398046, 398050, 397960
2	318759	149979	149985	149976
3	404107	199498	199805, 199805, 199807, 199808, 199807	199498, 199498
4	570363	290547	290547, 290547, 290547, 290547	290547
5	80417	47035	47036, 47035, 47035, 47034, 47036	47036, 47035
6	376968	51809	51813	50209

Table 8.2: Number of samples that exceed 3 s interval for each house

The figure below illustrates the measured samples for lighting which exceed the 3 sec sampling period.

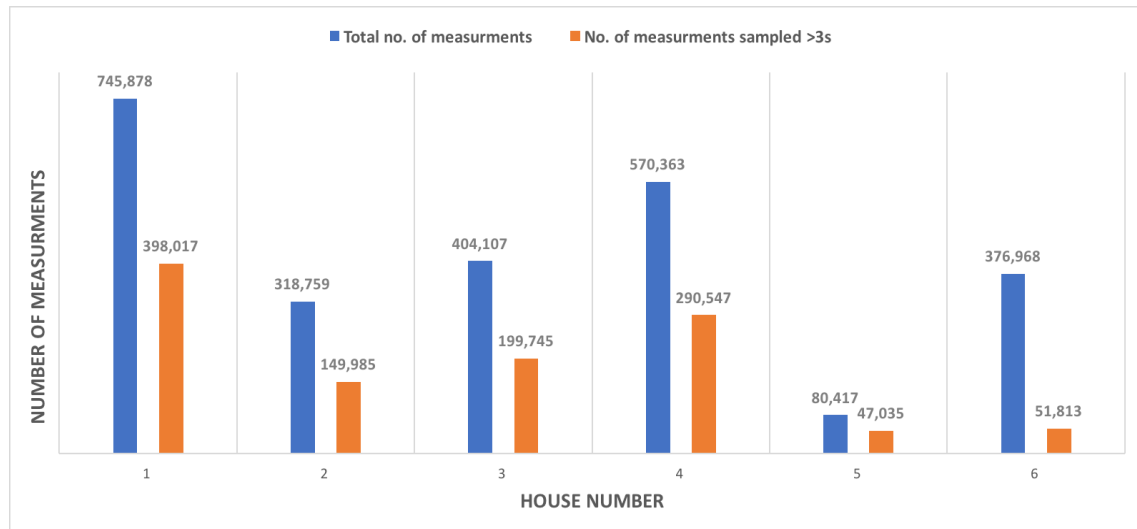


Figure 8.1: Measurement samples from lighting

The figure below illustrates the measured samples for washer dryer which exceed the 3 sec sampling period.

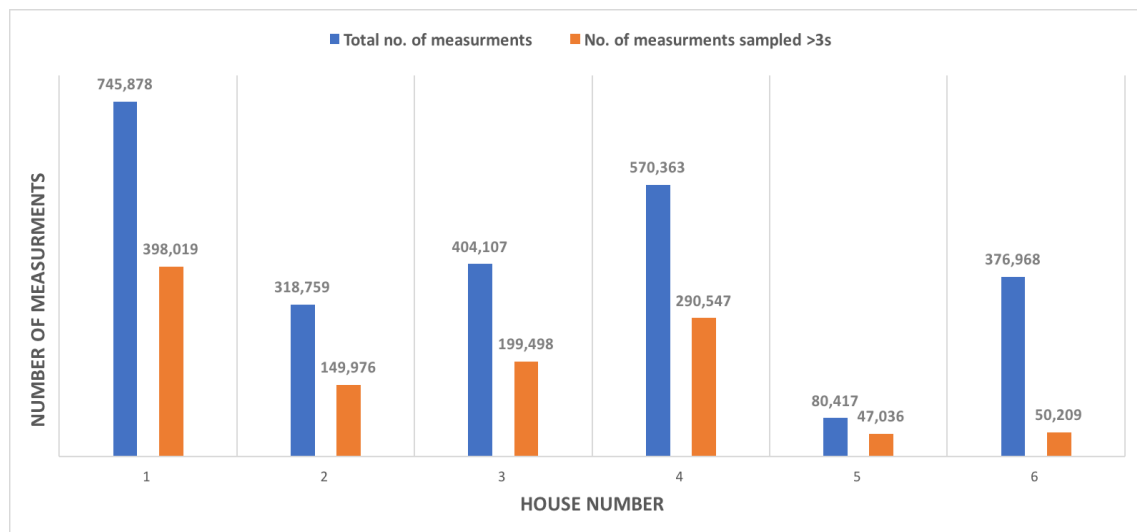


Figure 8.2: Measurement samples from washer dryer

9 Appendix B

This appendix contains the experimental results for choosing Adadelata as the optimizer for implementing the NN models.

The figure below illustrates the performance (loss) of various optimizers tested during the compilation of the models, resulting in the choice of using Adadelata.

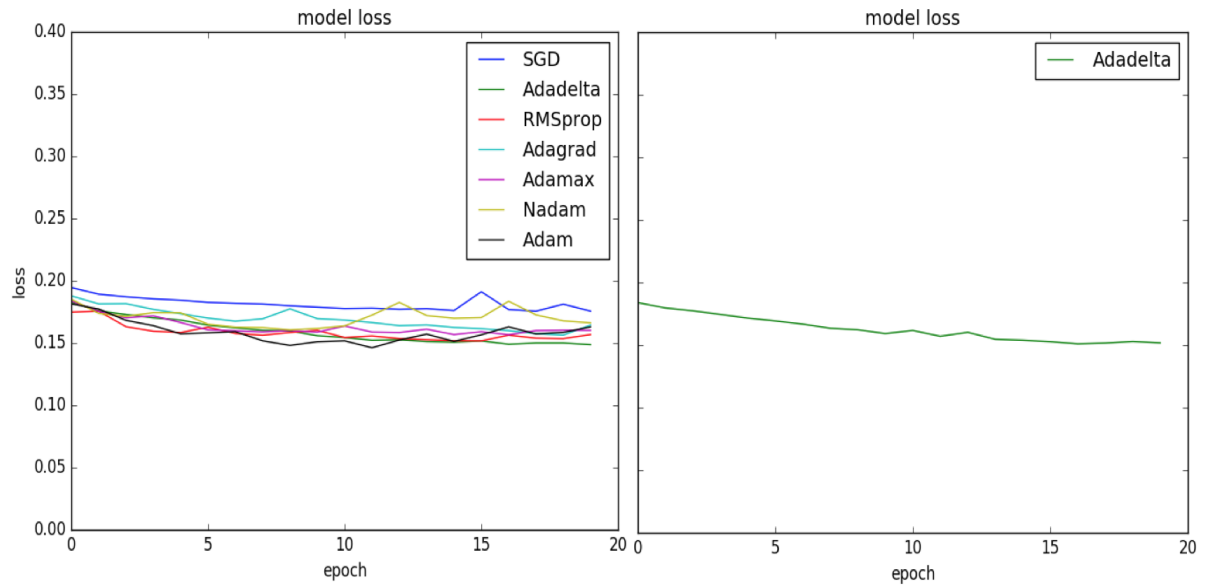


Figure 9.1: Optimizer effect on loss when window size is 2 hr for 20 epochs

The figure below illustrates the performance (accuracy) of various optimizers tested during the compilation of the models, resulting in the choice of using Adadelata.

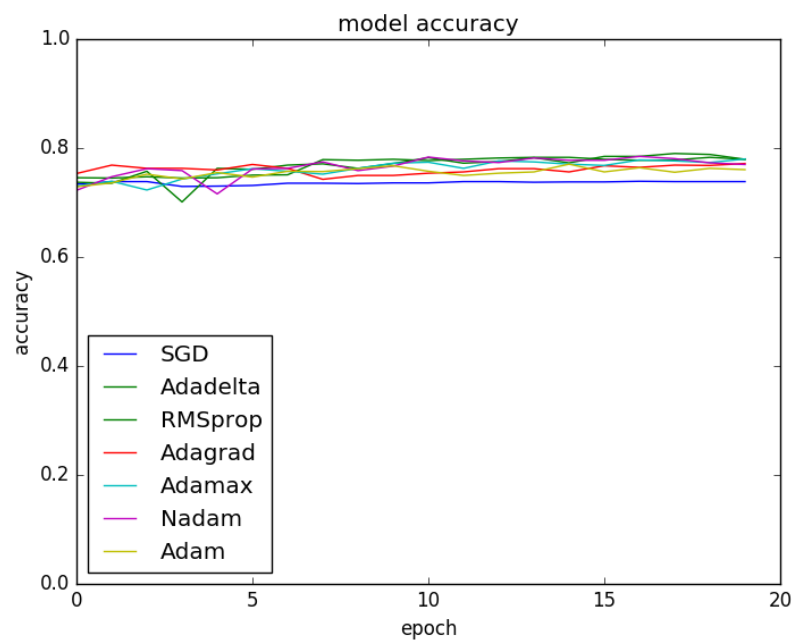


Figure 9.2: Effect of Optimizer on test accuracy for MLP approach of window size 2 hr

10 Appendix C

This appendix contains information about the CD structure which contains an electronic version of the thesis and the code implementation.

```
├─ Amrit_Raj_2159114_Bachelor_Thesis.pdf
├─ Thesis-Implementation
│   ├── Data
│   │   ├── X-Input
│   │   └── Y-Output
│   └── Models
│       ├── CNN-outlets-known.py
│       ├── CNN-outlets-unknown.py
│       ├── MLP-outlets-known.py
│       └── MLP-outlets-unknown.py
```

Figure 10.1: Folder structure in provided CD

The electronic version of the thesis is labelled "Amrit_Raj_2159114_Bachelor_Thesis".

The main folder "Thesis-Implementation" contains two subfolders "Data" and "Models".

The "Data" folder consists two subfolders. The "X-Input" folder has the data for dishwasher, lighting, washer dryer and unknown appliances across the 6 houses. The "Y-Output" folder consists of the mapped Y values of each window size as csv files.

The "Models" folder consists of the CNN and MLP python implementation for known and unknown appliances. To run the files the associated libraries need to be installed. These libraries needed are listed as imports in the python files. The files consists of the 8/18/8 model architecture. To obtain the other 4 architectures simply remove the layer and change the number of neurons to match the specifications. To run the python file, in the command line run "python <window size> <optimizer no.> <loss value>" Where the value of window size argument can be 900, 9604, 28900 and 86436. The values for the optimizer no. argument can be 0,1,2,3,4 and 5 corresponding to Adagrad, RMSprop, Adadelta, Adam, Adamax and SGD. The loss value argument can accept the arguments "mse" for mean squared error and "categorical_crossentropy" for categorical crossentropy.

Bibliography

- [1] S. Andrej Karpathy. Cs231n convolutional neural networks for visual recognition, 2017.
- [2] C. Y. F. Y. M. C. S. A. C. A. M. A. H. B. H. T. W. S. T. Andrew Ng, Jiquan Ngiam. Softmax regression, may 2017.
- [3] J. S. Armstrong and F. Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International journal of forecasting*, 8(1):69–80, 1992.
- [4] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.
- [5] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava. NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring. *ArXiv e-prints*, Apr. 2014.
- [6] Y. Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.
- [7] C. Chatfield and M. Yar. Holt-winters forecasting: Some practical issues. *The Statistician: Journal of the Institute of Statisticians*, 37:129–140, 1988.
- [8] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.
- [9] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.
- [10] M. Computing. Noise reduction and isolation, 2016.
- [11] J. T. Connor, R. D. Martin, and L. E. Atlas. Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2):240–254, 1994.
- [12] S. Darby et al. The effectiveness of feedback on energy consumption. *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, 486(2006), 2006.

- [13] U. A. de Madrid. Rule-based systems.
- [14] L. Deng and D. Yu. Deep learning: Methods and applications, May 2014.
- [15] T. G. Dietterich. Machine learning for sequential data: A review.
- [16] P. P. M. do Nascimento. Master thesis: Applications of deep learning techniques on nilm, 2016.
- [17] K. Ehrhardt-Martinez, K. A. Donnelly, and J. A. Laitner. Advanced metering initiatives and residential feedback programs: A meta-review for household electricity-saving opportunities, jun 2010.
- [18] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.
- [19] C. Fischer. Feedback on household electricity consumption: a tool for saving energy? *Energy efficiency*, 1(1):79–104, 2008.
- [20] G. Foody, M. McCulloch, and W. Yates. The effect of training set size and composition on artificial neural network classification. *International Journal of Remote Sensing*, 16(9):1707–1723, 1995.
- [21] H. S. Geller. Energy-efficient residential appliances: Performance issues and policy options. *IEEE Technology and Society Magazine*, 5(1):4–10, March 1986.
- [22] Z. Ghahramani, M. I. Jordan, and P. Smyth. Factorial hidden markov models. *Machine learning*, 29(2-3):245–273, 1997.
- [23] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.
- [24] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Aistats*, volume 15, page 275, 2011.
- [25] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [26] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.
- [27] G. W. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, Dec 1992.
- [28] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

- [29] D. O. Hebb. *The organization of behavior: A neuropsychological theory*. Wiley, New York, June 1949.
- [30] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012.
- [31] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.
- [32] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.
- [33] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- [34] B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050, 2014.
- [35] M. Humphries. Missing data & how to deal: An overview of missing data, 2016.
- [36] J. D. Hunter. Matplotlib, 2017.
- [37] image net.org. Large scale visual recognition challenge 2010 (ilsvrc2010), 2010.
- [38] T. Isokawa, H. Nishimura, and N. Matsui. Quaternionic multilayer perceptron with local analyticity. *Information*, 3(4):756–770, 2012.
- [39] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.
- [40] M. J. Johnson and A. S. Willsky. The hierarchical dirichlet process hidden semi-markov model. *CoRR*, abs/1203.3485, 2012.
- [41] M. J. Johnson and A. S. Willsky. Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research*, 14(Feb):673–701, 2013.
- [42] J. Kelly. Disaggregation of domestic smart meter energy data, 2016.
- [43] J. Kelly and W. Knottenbelt. The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes. *Scientific data*, 2, 2015.
- [44] Keras. Dense, may 2017.
- [45] Keras. Initializers, may 2017.

- [46] keras.io. Activations, 2017.
- [47] Keras.io. Convolutional layers, 2017.
- [48] keras.io. Optimizers, 2017.
- [49] Keras.io. The sequential model api, 2017.
- [50] J. Z. Kolter and T. S. Jaakkola. Approximate inference in additive factorial hmms with application to energy disaggregation. In *AISTATS*, volume 22, pages 1472–1482, 2012.
- [51] J. Z. Kolter and M. J. Johnson. Redd: A public data set for energy disaggregation research, 2011.
- [52] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [53] Lecture. Cnn tutorial.
- [54] Lecture. Why multilayer perceptron/neural network?
- [55] Lecture. Ics273a: Machine learning winter 2008 lecture 8, feb 2008.
- [56] H. Lecture. Cs281: Advanced machine learning, nov 2013.
- [57] S. Lecture. Lecture 19: Decision trees, nov 2014.
- [58] S. Lecture. Self taught learning (unsupervised learning), 2016.
- [59] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
- [60] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [61] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [62] Y. LeCun and C. Cortes. The mnist database of handwritten digits, 1998.
- [63] L. Library. Objective functions, sep 2015.
- [64] S. Mallat. Understanding deep convolutional networks. *Phil. Trans. R. Soc. A*, 374(2065):20150203, 2016.
- [65] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [66] W. McKinney. Pandas: Python data analysis library, 2017.

- [67] E. F. (MIT). Lecture 21: Intro to hidden markov models the baum-welch algorithm, 2010.
- [68] 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.
- [69] N. Murata, S. Yoshizawa, and S. Amari. Network information criterion-determining the number of hidden units for an artificial neural network model. *IEEE Transactions on Neural Networks*, 5(6):865–872, Nov 1994.
- [70] U. Nations. Paris agreement - status of ratification, 2017.
- [71] A. Ng. Cs229 lecture notes: Supervised learning, sep 2012.
- [72] U. D. of Energy. Estimating appliance and home electronic energy use.
- [73] S. K. Pal and S. Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on neural networks*, 3(5):683–697, 1992.
- [74] G. Panchal, A. Ganatra, Y. Kosta, and D. Panchal. Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers. *International Journal of Computer Theory and Engineering*, 3(2):332, 2011.
- [75] K. Prof. S. Sengupta, Indian Institute of Technology. Lec-1 introduction to artificial neural networks, dec 2003.
- [76] J. D. Rennie. Bounded loss classification, 2003.
- [77] G. repository: Keras. Mnist example, may 2017.
- [78] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
- [79] S. Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.
- [80] R. K. Samer Hijazi and C. Chris Rowen, IP Group. Using convolutional neural networks for image recognition.
- [81] scikit learn version 0.18.1. Train test split, may 2017.
- [82] F. Seide, G. Li, and D. Yu. Conversational speech transcription using context-dependent deep neural networks. In *Interspeech 2011*. International Speech Communication Association, August 2011.
- [83] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

- [84] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [85] O. Source. A beginner's guide to recurrent networks and lstms, 2016.
- [86] O. Source. Keras:mlp mnist approach, 2016.
- [87] O. Source. Keras: The python deep learning library, 2017.
- [88] O. Source. Lasagne objectives, 2017.
- [89] O. Source. Numpy, 2017.
- [90] C. Stanford. Cs231n convolutional neural networks for visual recognition, 2017.
- [91] C. State. History of the perceptron, online lecture, 2017.
- [92] R. K. M. A. Syrine Ben Driss, M Soua. A comparison study between mlp and convolutional neural network models for character recognition. *SPIE Conference on Real-Time Image and Video Processing*, 2017.
- [93] E. Systems. Powerport specifications, 2017.
- [94] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.
- [95] L. A. K. S. University. Cs231n convolutional neural networks for visual recognition, 2017.
- [96] V. U. o. T. Uwe Egly. Rule-based systems.
- [97] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.
- [98] R. Viola. New standard for smart appliances in the smart home. <https://ec.europa.eu/digital-single-market/en/blog>, dec 2015.
- [99] wiki. Nilm datasets, dec 2016.
- [100] D. Williams and G. Hinton. Learning representations by back-propagating errors. *Nature*, 323(6088):533–538, 1986.
- [101] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.
- [102] M. Zeifman and K. Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE Transactions on Consumer Electronics*, 57(1):76–84, February 2011.