COPPE
Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
UFRJ

NON-INTRUSIVE INDUSTRIAL LOAD MONITORING ON A FACTORY IN BRAZIL

Pedro Bandeira de Mello Martins

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: José Gabriel Rodriguez Carneiro Gomes

Rio de Janeiro
Outubro de 2020

NON-INTRUSIVE INDUSTRIAL LOAD MONITORING ON A FACTORY IN BRAZIL

Pedro Bandeira de Mello Martins

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientador: José Gabriel Rodríguez Carneiro Gomes

Aprovada por: Prof. José Gabriel Rodríguez Carneiro Gomes
Prof. Mariane Rembold Petraglia
Prof. Charles Bezerra do Prado

RIO DE JANEIRO, RJ – BRASIL
OUTUBRO DE 2020

Martins, Pedro Bandeira de Mello

Non-intrusive industrial load monitoring on a factory in Brazil/Pedro Bandeira de Mello Martins. – Rio de Janeiro: UFRJ/COPPE, 2020.

XIV, 83 p.: il.; 29, 7cm.

Orientador: José Gabriel Rodriguez Carneiro Gomes

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2020.

Referências Bibliográficas: p. 48 – 54.

1. Deep Learning. 2. Non-intrusive load monitoring. 3. Energy Efficiency. 4. Data set. I. Gomes, José Gabriel Rodriguez Carneiro. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

To my mother.

Acknowledgement

I wish to show my appreciation for GreenAnt[1] and all my co-workers including Vagner Nascimento, Pedro Bittencourt, Raphael Guimarães, who let me research this project (PD-00386-1606/2016) and use it on my master thesis. I would like to give a special thanks to Rafael Rocha, who traveled many times to Minas Gerais to install GreenAnt's meters in the factory; to Rodrigo Fructuoso and Caio Mehlem, who created the meter's hardware; and Eduardo Morgan, who helped me build many insights regarding this research project.

I would like to express my gratitude to my advisor, Prof. José Gabriel Rodríguez Carneiro Gomes, who guided me through this project and who was always ready to answer my questions. I wish to extend my thanks to the Laboratory for the Processing of Analog and Digital Signals (PADS) for all the infrastructure it provided me to progress this project. I cannot forget to thank my fellow colleagues and friends Leonardo Mazza and Olavo Sampaio, who spent countless hours studying with me in PADS after work.

Finally, I wish to acknowledge my family for all the help they gave me since my childhood. Without my mother, Paula Bandeira, aunt Carla Bandeira and grandmother Heloisa Bandeira, I would certainly not be able to do all the needed steps to defend this master thesis. I must thank Luiza Conde, as she helped me tremendously when I was feeling the most stress and anxiety during this year due to COVID-19 pandemic, work, and thesis writing.

[1] https://www.greenant.com.br/english/greenant

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

MONITORAMENTO NÃO INTRUSIVO DE CARGAS INDUSTRIAIS EM UMA FÁBRICA NO BRASIL

Pedro Bandeira de Mello Martins

Outubro/2020

Orientador: José Gabriel Rodriguez Carneiro Gomes

Programa: Engenharia Elétrica

Esta dissertação aborda uma aplicação e comparação de um conjunto de técnicas de Non-Intrusive Load Monitoring (Monitoramento Não-Intrusivo de Cargas, NILM) em dados elétricos coletados de uma fábrica no Brasil. NILM propõe determinar o consumo de energia de um único aparelho a partir da demanda total dos consumidores sem a necessidade de instalação de sensores intrusivos ou mais de um medidor por quadro de energia. Como o foco principal desta tese é estudar NILM em ambientes industriais e até a data da escrita nenhum dado público disponível foi encontrado, um conjunto de dados (IMDELD) foi criado para esta pesquisa em uma fábrica de ração avícola usando medidores inteligentes. IMDELD possui um total de onze diferentes classes de assinaturas elétricas, incluindo oito classes de máquinas industriais, dois diferentes subcircuitos e um circuito principal. Os dados foram coletados em uma frequência de 1 Hz por até cento e onze dias.

Para atingir este objetivo da comparação de métodos, dois métodos são implementados: Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov, FHMM) e Deep Learning (WaveNILM). Em paralelo com os modelos FHMM, os modelos baseados no Deep Learning têm menor Signal Aggregated Error (Erro Agregado de Sinal, SAE) e Normalized Disaggregation Error (Erro de Desagregação Normalizado, NDE). De acordo com a F_1-Score (Medida F1, F1), eles também identificaram aparelhos individuais ligados ou desligados em uma porcentagem maior do tempo testado.

Dentre todas as máquinas, WaveNILM atingiu F1 0.93 ± 0.07, enquanto FHMM pontuou F1 0.79 ± 0.12. WaveNILM predice máquinas com SAE médio 0.1 ± 0.2 e NDE médio 0.1 ± 0.2, enquanto FHMM predice máquinas com SAE médio 0.3 ± 0.2 e NDE médio 0.3 ± 0.2.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

NON-INTRUSIVE INDUSTRIAL LOAD MONITORING ON A FACTORY IN BRAZIL

Pedro Bandeira de Mello Martins

October/2020

Advisor: José Gabriel Rodriguez Carneiro Gomes

Department: Electrical Engineering

This thesis addresses the comparison of two techniques of Non-Intrusive Load Monitoring applied to electrical data collected from a factory in Brazil. NILM proposes to separate single-appliance power consumption from consumers total demand without the need for installation of intrusive sensors or more than one meter per building. As the main focus of this thesis is to study NILM on industrial settings and, until the date of writing, no public data were found, IMDELD data set was collected for this research on a poultry feed factory using smart meters. IMDELD has a total of eleven classes of electrical signatures, including eight classes of heavy-industry machines, two different sub-circuits, and a main circuit. The data was collected at a 1 Hz rate for up to a hundred eleven days.

To achieve the comparison goal, two methods are implemented: Factorial Hidden Markov Models and Deep Learning (WaveNILM). In comparison to the FHMM models, the Deep Learning-based models have smaller Signal Aggregated Error and Normalized Disaggregation Error. They also identified single-appliances as turned ON or OFF on a larger percentage of the time tested based on F_1-Score.

Among all appliances, on average WaveNILM F1-scored 0.93 ± 0.07 while FHMM scored 0.79 ± 0.12. WaveNILM predicted machines with average SAE 0.1 ± 0.2 and NDE 0.1 ± 0.2, while FHMM predicted machines with average SAE 0.3 ± 0.2 and NDE 0.3 ± 0.2.

Contents

List of Figures

List of Tables

Acronyms

AFHMM Additive Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov Aditivos)

ANN Artificial Neural Networks

ART Adaptative Ressonant Theory

CNN Convolutional Neural Networks

CPU Central Processing Unit

CT Current Transformer

CVD Continuously Varying Devices

DC Direct Current

DPI Double-pole contactor I

DPII Double-pole contactor II

DT Decision Trees

EFI Exhaust Fan I

EFII Exhaust Fan II

F1 F_1-Score (Medida F1)

FHMM Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov)

FNN Feedforward Neural Networks

FSM Finite-State Machines

GASM-B GreenAnt Smart Meter Beta

GB Gigabytes

GPU Graphical Processing Unit

GRU Gated Recurrent Unit

GSP Graph Signal Processing

HMM Hidden Markov Models

IMDELD Industrial Machines Dataset for Electrical Load Disaggregation

KNN K-Nearest Neighbours Algorithm

LSTM Long Short-Term Memory

LVDB Low Voltage Distribution Board

MI Milling Machine I

MII Milling Machine II

MAE Mean Absolut Error

MLP Multilayer Perceptron

MV/LV Medium-to-Low Voltage Transformer

NDE Normalized Disaggregation Error (Erro de Desagregação Normalizado)

NILM Non-Intrusive Load Monitoring (Monitoramento Não-Intrusivo de Cargas)

PI Pelletizer I

PII Pelletizer II

PCD Permanent Consumer Devices

RAM Random-Access Memory

RMS Root-mean squared

RMSE Root Mean Square Error

RNN Recurrent Neural Networks

SAE Signal Aggregated Error (Erro Agregado de Sinal)

SGD Stochastisc Gradient Descent

TCN Temporal Convolutional Networks

THD Total Harmonic Distortion

Chapter 1

Introduction

This work presents an implementation of a temporal convolution-based deep neural network to solve load disaggregation on Non-Intrusive Load Monitoring (Monitoramento Não-Intrusivo de Cargas, NILM) problem for a data set [1] collected on a factory in Minas Gerais, Brazil. The main model here presented is compared to a Factorial Hidden Markov Model previously used on industrial sites.

NILM, a term coined by Hart in 1992 [2], refers to the monitoring of individual appliance energy consumption of a desired site without the extensive use of intrusive meters or appliance-specific sensors. Load disaggregation – sometimes referred to as a synonym to NILM – is the specific method for disaggregation of the single-appliances load from a whole-building energy consumption signal.

1.1 Objective

The main objectives of this work are: (1) to present a data set of heavy-machinery energy from a Brazilian factory, (2) to infer energy consumption of industrial machinery using energy data from one meter at the factory Medium-to-Low Voltage Transformer (MV/LV), (3) to present a new model of disaggregation based on Temporal Convolutional Networks (TCN), (4) to compare the aforementioned model with Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov, FHMM) model found in the literature using the presented data set.

1.2 Constraints

Eleven meters are used to collect energy data from a factory in Brazil. Out of those, one meter is installed for whole-building energy consumption data acquisition, and the other ten are installed at sub-circuits or machines to collect data for supervised training. Root-mean squared (RMS) voltage, RMS current, active power, reactive power, apparent power and active energy are sampled at 1 Hz for each meter. The time series created is then divided into windows of 1024 samples. All models are trained using part of those collected windows and tested using the remaining data.

The models are implemented in Python, using libraries such as SciPy [3], Numpy [4], Pandas [5], TensorFlow [6], Keras [7] and NILMTK [8]. The models are trained in a machine with 20 GB of RAM, a GPU NVidia GTX1060 with 6 GB RAM and an Intel(r) Core(TM) i5-440 CPU @ 3.10 GHz with four cores.

1.3 Motivation

The 21st century alone has seen an increase of 34.17% of electric power consumption per population (kWh/capita) worldwide [9]. Efficient energy use can help slow down this trend and increase the energy supply without the need for more power generation, therefore reducing both costs for new infrastructure and greenhouse gas emissions. Consumer decision making is an essential aspect of saving energy and, thus increasing energy usage efficiency. FREDERIKS *et al.* [10] explain how consumers do not decide about their electrical energy consumption rationally, giving an insight of how it is vital to give all information possible to them, so they can check their behaviors and possibly decrease their electricity costs.

Real-time information of energy consumption down to appliance level credit up to 20% of energy savings according to ARMEL *et al.* [11] and DARBY *et al.* [12]. ZEIFMAN and ROTH [13] cite benefits of appliance-specific consumption information such as fault detection, behavioral pattern elucidation, appliance analysis based on use, better data for energy-aware appliance redesign, improved load forecasting, improved economic models, and label efficiency changes. NILM aims at prospecting information of this kind without the need for several meters, and it has been mainly studied in residential and commercial buildings [14–21].

Although the industrial sector corresponds to 31.55% of worldwide electrical energy supply [9], industry-related NILM is still not widely researched [22, 23]. In Brazil, the industrial sector consumes up to 33% [24] of the total electricity consumption; therefore, the use of energy efficiency programs on industrial plants could benefit the entire national grid. Insights granted from NILM inferences can help with energy wastefulness, with energy quality, and they can leverage demand response on the sector. NILM gives meaningful feedback not only to consumers, but also to energy system operators, who can better predict how their customers will behave, and can then offer focused services based on machines disaggregated from their consumption.

1.4 Description

The following chapters treat in more detail the problem addressed in this work. The first section of Chap. 2 intends to introduce and briefly explain what NILM is and its primary literature. After the first section, Chap. 2 aims at explaining the main concepts of both HMM and ANN with a later focus on Temporal Convolutional Networks deep learning models.

Chapter 3 gives more details about the data set, including some background on the factory process and measurement points. After that, the text explains how the models use the data set and how it is processed.

Chapter 4 has the main objective of explaining how the models are implemented and trained. Chapter 5 analyses the results obtained from the experiments and Chap. 6 presents conclusions and proposes future work ideas.

Chapter 2

Theory

This chapter covers the main theoretical foundations required for this dissertation. It is divided into three sections; Sec. 2.1 aims at introducing NILM and Load Disaggregation with its basic concepts, technical issues, disaggregation methods, learning methods and needed data set properties. Section 2.2 refers to ANN, Deep Learning and temporal convolution models. Furthermore, Sec. 2.3 explains HMM.

2.1 Non-Intrusive Load Monitoring

NILM cannot be regarded as a solved problem, even though the concept was proposed in 1992 [2]. Also known as load disaggregation, it is a process of blind source separation. Generally speaking, we cannot identify how many components were summed to generate the aggregate signal. This creates an ill-posed problem, and a series of statistical models and algorithms are used to approximate the original components. A good recent review can be found in [25].

2.1.1 Problem definition

A whole-building electricity consumption signal measured by a meter can be formulated as in Eq. (2.1):

$$\begin{aligned} \mathbf{x}_t &= \sum_{n=1}^{N} \mathbf{x}_{n,t} + \mathbf{r}_t \\ \mathbf{x}_t &= g\left(\mathbf{x}_{1,t}, \mathbf{x}_{2,t}, \mathbf{x}_{3,t}, \cdots\right) \end{aligned} \tag{2.1}$$

where N is the number of appliances, $\mathbf{r}_t$ is an intrinsic noise at a given time t, $\mathbf{x}_{n,t}$ is the consumption signal vector of the appliance n at t, and $\mathbf{x}_t$ is the aggregated signal vector at t. The measurement $\mathbf{x}$ is defined as a vector, because it can have multiple

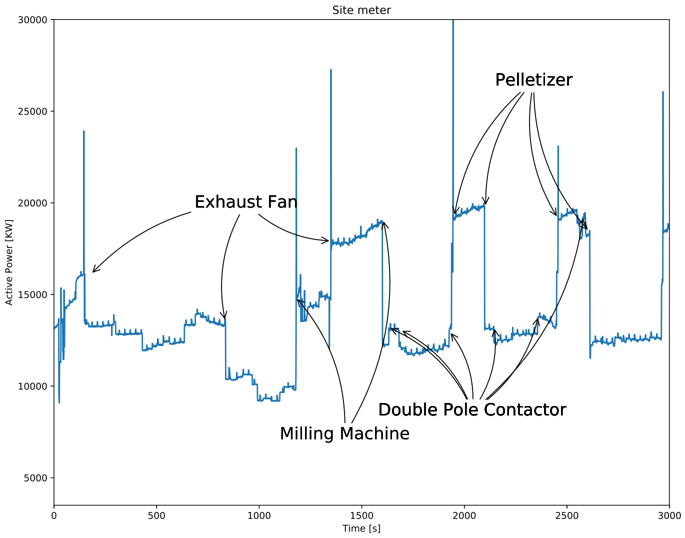


Figure 2.1: Illustration of a whole-building power signal with appliances events annotated. (Reprinted with permission from MARTINS *et al.* [23])

dimensions, such as electric potential difference (V), electric current (I), active power (P), reactive power (Q), apparent power (S), Total Harmonic Distortion (THD) or any other feature measured. Figure 2.1 illustrates an example of a whole-building power signal with appliances events annotated.

NILM challenge is to find a function $f(\cdot)$ such that given $\mathbf{x}_t$, $f(\mathbf{x}_t)$ results in $\mathbf{x}_{1,t}, \mathbf{x}_{2,t}, \mathbf{x}_{3,t}, \cdots$. Although NILM is an inverse problem, g is not a bijective function; therefore f is not an inverse function of g and f has no unique solution. Even if $\mathbf{r}_t = \mathbf{0}$ the system in Eq. (2.1) is an underdetermined system, since there are more variables than equations. Different combinations of $\mathbf{x}_{n,t}$ could result in the same $\mathbf{x}_t$. As this problem is ill-posed, we try to find a function $\tilde{f}$ that returns an approximate solution $\tilde{\mathbf{x}}_{n,t}$ for $\mathbf{x}_{n,t}$. If we define a loss function $\mathcal{L}$, the feasible objective of NILM is to find $\tilde{f}$ such that $\mathcal{L}(\tilde{\mathbf{x}}_{n,t} - \mathbf{x}_{n,t}) \to 0$, as seen in Eqs. (2.2) and (2.3).

$$\tilde{f}(\mathbf{x}_t) = \begin{bmatrix} \tilde{\mathbf{x}}_{1,t} \\ \tilde{\mathbf{x}}_{2,t} \\ \tilde{\mathbf{x}}_{3,t} \\ \vdots \end{bmatrix} \tag{2.2}$$

$$\mathcal{L}(\tilde{\mathbf{x}}_{n,t} - \mathbf{x}_{n,t}) \to 0 \tag{2.3}$$

2.1.2 Appliance modeling

Appliances are modeled in three ways: (1) Finite-State Machines (FSM) when an appliance can be a simple two-state (ON/OFF) machine, or a multistage machine, for example, washing machines; (2) Continuously Varying Devices (CVD) when an appliance does not have quantized states, which is equivalent to say that its power consumption varies over time, for example, a desktop computer; and (3) Permanent Consumer Devices (PCD) when an appliance has constant power consumption, for example, a traffic light. For an FSM modeled appliance, $\mathbf{x}_{n,t}$ from Eq. (2.1) can be defined as in Eq. (2.4):

$$\tilde{\mathbf{x}}_{n,t} = \sum_{k=1}^{K_n} z_{n,t,k} \boldsymbol{\mu}_{n,k} \tag{2.4}$$

as it models each k-th state feature vector $\boldsymbol{\mu}_{n,k}$ of all K_n states of the appliance n. $z_{n,k,t}$ indicates whether the state k corresponds to operation or not. There can only be one state in operation at a given time t, that is to say $\sum_{k=1}^{K_n} z_{n,t,k} = 1$ and $z_{n,t,k} \in \{0, 1\}$. CVD and PCD $\mathbf{x}_{n,t}$ formulas are given in Eqs. (2.5) and (2.6) respectively.

$$\tilde{\mathbf{x}}_{t,n} = \boldsymbol{\mu}_n(t) \tag{2.5}$$

$$\tilde{\mathbf{x}}_{t,n} = \boldsymbol{\mu}_n \tag{2.6}$$

Using appliance-specific models can create some problems in deciding which solver should be used, as will be seen in Sec. 2.1.3. Each solution asks for a particular modeling approach. There is no need to model each appliance differently on a building. We can generalize or constrain every appliance to be under one of the previous models.

2.1.3 Event-based and eventless approaches

Solving NILM can be divided into two main approaches: event-based and eventless. Event-based approaches were first introduced by HART [2] and remained as the main focus of research in NILM until recently. Those approaches rely on event detection, feature extraction and load identification to estimate each appliance energy consumption under a given total consumption. Figure 2.2 illustrates this process.

Event detection searches for features that indicate when an appliance changes its state. Those features are usually associated with high-frequency events and switches from a steady-state. Event detectors can be implemented, among other techniques, using: heuristics [2, 26], probabilistic models [27], matched filters [28], Decision

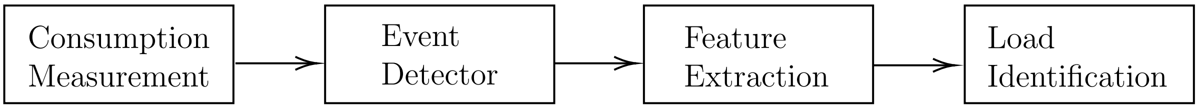


Figure 2.2: Block diagram of an event-based NILM algorithm.

Trees (DT), and Long Short-Term Memory (LSTM) [29]. ANDERSON *et al.* [30] review event detectors methods for NILM and propose metrics to evaluate them.

Load identification refers to the classification, using features related to a detected event, of a particular appliance. A myriad of methods can be applied here, including supervised and unsupervised. We can cite as examples of methods used on load identification K-Nearest Neighbours Algorithm (KNN) [31], Graph Signal Processing (GSP) [17, 28], Adaptative Ressonant Theory (ART) [26] and probabilistic knapsack algorithms [32].

Eventless approaches do not use event detectors nor feature extractors, thus simplifying the NILM process. In this context, a consumption measurement is directly input into a load identification algorithm. Figure 2.3 illustrates a simple block diagram of an eventless approach.

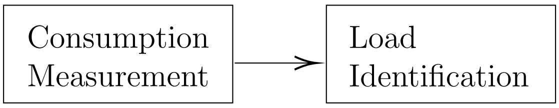


Figure 2.3: Block diagram of an eventless NILM algorithm.

Using eventless approach, PARSON *et al.* [15] propose a HMM-base solution. They model each appliance as an FSM, assuming that those states are non-observable and can be modeled as an HMM. In an HMM only the output is observable, so its states are hidden and their probability functions depend on the output. This method requires that every appliance is modeled as a Markov state and assumes that every step change in aggregate power is due to a change in a Markov state. Therefore this model does not acknowledge if two states are changed at the same time, which can be a strong assumption. There are groups of appliances that are turned on together, especially on factories (for example pelletizers and exhaust-fans).

KOLTER e JAAKKOLA [16] try to address this issue by using Additive Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov Aditivos, AFHMM). This method considers every aggregate consumption data point as a result of an additive function of different hidden states, each state being assigned to one appliance, similar to Eq. (2.1) using Eq. (2.4). Each HMM inside an AFHMM is independent and can change state simultaneously with another HMM. This approach presents some difficulties, as the exact inference of the AFHMM is highly susceptible to local optima, so the authors assume that there is a mixture component that cannot be modeled, and they use approximate inference with the constrain that each state can not change simultaneously. This model was also applied on a previous load disaggregation research on the industrial sector [22]. HOLMEGAARD e KJAERGAARD [22] achieve better results in a cold store using a day-specific FHMM, as this specific industrial site had different consumption patterns depending on the day. This does not happen in IMDELD, so a normal FHMM is used.

KELLY e KNOTTENBELT [14], DO NASCIMENTO [20], ZHANG *et al.* [18] and MORGAN [21] use Deep Learning techniques to estimate residential appliance loads based only on aggregate consumption samples. They achieve better results and higher generalization power in comparison with HMM. ANN and Deep Learning techniques will be discussed in Sec. 2.2. This work discusses the use of a Deep Learning model and compares it with HMM in an industrial sector load disaggregation problem.

2.2 Artificial Neural Networks

ANNs are a set of mathematical models inspired by a biological neural network. Like their biological counterpart, they are composed of neurons and their connections. A signal is transmitted and transformed through the network, which improves through a learning process. This section presents basic concepts of ANNs, how a training phase occurs, and a summary of ANNs in time series.

An artificial neuron computes a dot product of an input vector $\mathbf{x} \in \mathbb{R}^{\mathrm{N}}$, having N features, and a weight vector $\mathbf{w} \in \mathbb{R}^{\mathrm{N}}$. A bias b is added to this result and then the results is fed through an activation function $g : \mathbb{R}^{\mathrm{N}} \rightarrow \mathbb{R}$ which returns an output $o \in \mathbb{R}$. Equation (2.7) represents this procedure. Those neurons can be arranged in different ways, and one example is a Feedforward Neural Networks (FNN), in which the neurons are arranged in sequenced layers. A layer receives the output of the previous layer as an input.

$$o = g\left(\mathbf{w}^T \mathbf{x} + b\right) \tag{2.7}$$

$$\mathbf{o} = G\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right) \tag{2.8}$$

Equation (2.8) represents a neuron layer, where $\mathbf{o} \in \mathbb{R}^{\mathrm{M}}$ is the output vector of all M neurons in the layer, $\mathbf{W} \in \mathbb{R}^{\mathrm{M \times N}}$ is the matrix of weights and $\mathbf{b} \in \mathbb{R}^{\mathrm{M}}$ is the bias vector. G is an activation function $G : \mathbb{R}^{\mathrm{M}} \rightarrow \mathbb{R}^{\mathrm{M}}$. This layer is called fully-connected because, generally, all neurons are connected with nonzero weights $w_{n,m}$ to all inputs. FNN with multiple layers are called Multilayer Perceptron (MLP).

An intermediate layer of an MLP is called hidden layer, as its output is usually fed to another layer, and are not visible from the output. Figure 2.4 represents an MLP with two hidden layers $\mathbf{W}_{(1)}$, $\mathbf{W}_{(2)}$ and one output layer $\mathbf{W}_{(3)}$. Biases and activation functions are omitted inside the circles.

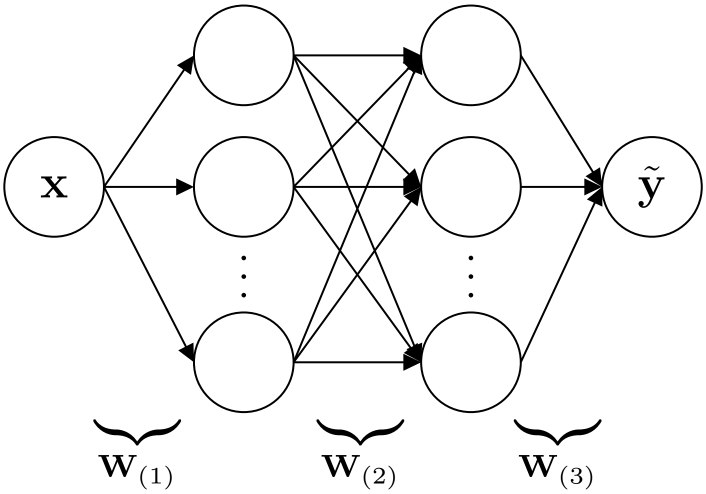


Figure 2.4: MLP with $G_{(3)}\left(\mathbf{W}_{(3)}G_{(2)}\left(\mathbf{W}_{(2)}G_{(1)}\left(\mathbf{W}_{(1)}\mathbf{x}+\mathbf{b}_{(1)}\right)+\mathbf{b}_{(2)}\right)+\mathbf{b}_{(3)}\right)=\tilde{\mathbf{y}}$ representation.

MLPs are universal approximators in the sense that they can approximate, with arbitrarily small error, any continuous function within the unitary hypercube $0, 1^N$ if and only if the activation function $G : \mathbb{R}^{\mathrm{M}} \rightarrow \mathbb{R}^{\mathrm{M}}$ is non-polynomial [33]. The model used in this work uses four kinds of activation functions: linear function, hyperbolic tangent (2.9), logistic function (2.10) and ReLU (2.11). In that way, with a given loss function $\mathcal{L}$, we can model a neural network that approximates a desired function f after a learning process. Although a single hidden layer is sufficient for an MLP to be an universal approximator [34], shallow networks (networks with a single hidden layer) need prohibitively more neurons than deep networks to approximate the same function [35].

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.9}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.10}$$

$$\mathrm{ReLU}(x) = \max(x, 0) \tag{2.11}$$

2.2.1 Training

ANNs are parametric functions $\phi_{\boldsymbol{\theta}}(\mathbf{x})$. The parameters $\boldsymbol{\theta}$ are all learnable variables, like weights and biases, and they are locally optimized, iteratively, during a training phase, which can be supervised or unsupervised. If we plan to approximate an unknown function f through an ANN, then we must find a set of parameters $\boldsymbol{\theta}$ that would describe a $\phi_{\boldsymbol{\theta}}(\mathbf{x}) \approx f(\mathbf{x})$. If the inputs $\mathbf{x}$ and the desired outputs $f(\mathbf{x}) = \mathbf{y}$ are known, then we can train the network using supervised learning.

Supervised learning is not the only possible way of optimizing $\boldsymbol{\theta}$ from an ANN. In some cases the available set of labelled input-output pairs is not enough to properly represent the real distribution, then a semi-supervised learning algorithm can be used. If there are no labelled $\mathbf{y}$ available on the data set, it is still possible to train an ANN using unsupervised learning. Even when supervised learning is used to optimize $\boldsymbol{\theta}$, an unsupervised learning can be used as a pre-training phase.

Optimization

A training phase in supervised learning aims at minimizing the expected value of a loss function $\mathcal{L}\left(\phi_{\boldsymbol{\theta}}\left(\mathbf{x}\right), \mathbf{y}\right)$, as seen in Eq. (2.12), by updating $\boldsymbol{\theta}$ using optimization methods. As it is extremely difficult – if not impossible – to have a data set of all possible labelled input-output pairs, an approximation of the expected value is used as seen in Eq. (2.13). The set $\left\{\left(\mathbf{x}_1, \mathbf{y}_1\right), \left(\mathbf{x}_2, \mathbf{y}_2\right), \cdots, \left(\mathbf{x}_N, \mathbf{y}_N\right)\right\}$ used during a training phase is called a training set.

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}} \left[\mathcal{L}\left(\phi_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{y}\right)\right] \tag{2.12}$$

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(\phi_{\boldsymbol{\theta}}(\mathbf{x}_n), \mathbf{y}_n\right) \tag{2.13}$$

This optimization problem is mostly differentiable, apart from some finite number of points, and is solved iteratively using Gradient Descent algorithms. As training sets are usually too large to be processed in one batch, Stochastisc Gradient Descent (SGD) algorithms are used, in which the data set is divided into mini-batches $\left\{\left(\mathbf{x}_1, \mathbf{y}_1\right), \left(\mathbf{x}_2, \mathbf{y}_2\right), \cdots, \left(\mathbf{x}_M, \mathbf{y}_M\right)\right\}$, where $M < N$ and each mini-batch is uniformly drawn from the training set. SGD algorithms update $\boldsymbol{\theta}$ iteratively as mini-batches are drawn.

Depending on the problem set up, different loss functions can be chosen as $\mathcal{L}$. In this work we use MAE (Eq. (2.14)), because it is one of the usual average model-performance errors alongside with Root Mean Square Error (RMSE) (Eq. (2.15)). As the name denotes, MAE is the average absolute difference over a sample. MAE $\in [0, \infty)$ and it does not account for error direction (whether an error is more negative or positive). This loss function is usually compared to RMSE, and interesting articles about it can be found in [36, 37].

$$\mathrm{MAE}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} |x_i - \tilde{x}_i| \tag{2.14}$$

$$\mathrm{RMSE}(\mathbf{x}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (x_i - \tilde{x}_i)^2} \tag{2.15}$$

2.2.2 ANN in Time series

Unlike static data, time series represent an infinite data stream which is temporarily auto-correlated. This type of data is used in several different areas, such as communications, weather forecast, seismic analysis, electrical demand forecast and others. ANNs can be used to forecast, analyse, regress, and generate a time series. This work generates appliance time series based on an unique whole-building time series.

In terms of input-to-output, an ANN model for sequential data can be defined in four different ways: one-to-one, one-to-many, many-to-one, and many-to-many models. A one-to-one model predicts a data point for each one it reads, while a one-to-many model will generate more than one data point for each input. Many-to-one and many-to-many models, on the other hand, use points in the vicinity to predict one or many points in the time series. Using features from a time instant after the current output timestamp configures a noncausal model, which can be undesirable.

Recurrent Neural Networks

It is possible to train an MLP model with a time series, but it does not necessarily acknowledge sequential correlation, and it only accepts fixed-sized vectors as inputs and outputs. Recurrent Neural Networks (RNN) address those issues with a state vector $\mathbf{h}_t$, which stores all previous inputs. The state vector equation is given by Eq. (2.16), where tanh is the hyperbolic tangent over all vector dimensions, $\mathbf{W}_{hh}$ is a learnable state weight matrix, and $\mathbf{W}_{hx}$ is a learnable input-to-state weight matrix. Bias vectors are not shown for simplicity.

$$\begin{aligned}\mathbf{h}_0 &= \mathbf{0}\\ \mathbf{h}_t &= \tanh\left(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{hx}\mathbf{x}_t\right)\end{aligned} \tag{2.16}$$

With Eq. (2.16), an RNN learns how important its previous inputs are and stores them. This is the main feature of an RNN and ensures that the model is able to learn the time series' temporal correlation. The weight matrices are initialized with random numbers, just like an MLP and are optimized via a backpropagation through time algorithm [38]. It is also possible to stack layers of RNN cells and create deep models, if activation functions that do not let gradients to vanish are used.

RNNs tend to have problems in learning when relevant information is too distant in time, as discussed by [39]. This happens because Gradient Descent becomes inefficient, it can oscillate, go to infinity or vanish, and the training phase is affected. To avoid this issue LSTM was introduced by [40]. This neural network introduces a new memory cell ($\mathbf{C}_t$) and three new gate units that help to forget ($\mathbf{f}_t$), store ($\mathbf{i}_t$), and propagate to the output $\mathbf{o}_t$ information. As seen in Eqs. (2.18) to (2.19), all of the gates use the same principle. They use learnable weight matrices, an input vector, and a state vector in order to output a number in between $(0, 1)$ along each feature dimension.

$$\mathbf{f}_t\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) = \sigma\left(\mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fx}\mathbf{x}_t\right) \tag{2.17}$$

$$\mathbf{i}_t\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) = \sigma\left(\mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ix}\mathbf{x}_t\right) \tag{2.18}$$

$$\mathbf{o}_t\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) = \sigma\left(\mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{ox}\mathbf{x}_t\right) \tag{2.19}$$

The new memory cell $\mathbf{C}_t$ stores hidden information through time in the LSTM cells. A forget gate $\mathbf{f}_t$ can erase this cell state, or diminish its importance through the cells, while an input gate $\mathbf{i}_t$ manages storage of new information into it. Equation (2.21) shows how a memory cell in an LSTM cell works. The symbol $\odot$ represents an element-wise multiplication.

$$\widetilde{\mathbf{C}}_t = \tanh\left(\mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{W}_{cx}\mathbf{x}_t\right) \tag{2.20}$$

$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \widetilde{\mathbf{C}}_T \tag{2.21}$$

The output of an LSTM cell is given by Eq. (2.22). The output gate $\mathbf{o}_t$ analyses the last output and the new input to set the output. More on the subject can be seen in [41].

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{C}_t\right) \tag{2.22}$$

Another variation of an RNN is called Gated Recurrent Unit (GRU) [42]. It uses two gates instead of three in the LSTM. Those gates are called update gate $\mathbf{u}_t$ and reset gate $\mathbf{r}_t$ (Eq. (2.24)). The update gate is used to decide how much of the older information will be retained, while the reset gate forgets older information if needed.

$$\mathbf{u}_t\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) = \sigma\left(\mathbf{W}_{uh}\mathbf{h}_{t-1} + \mathbf{W}_{ux}\mathbf{x}_t\right) \tag{2.23}$$

$$\mathbf{r}_t\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) = \sigma\left(\mathbf{W}_{rh}\mathbf{h}_{t-1} + \mathbf{W}_{rx}\mathbf{x}_t\right) \tag{2.24}$$

As seen in Eq. (2.26), the reset gate is used when computing a hidden output $\widetilde{\mathbf{h}}_t$, which has a desired amount of past information. The update gate is used to choose how much past information $\mathbf{h}_{t-1}$ or $\widetilde{\mathbf{h}}_t$ the output $\mathbf{x}_t$ has.

$$\widetilde{\mathbf{h}}_t = \tanh\left(\mathbf{W}_{hx}\mathbf{x}_t + \mathbf{r}_t \odot \mathbf{W}_{hh}\mathbf{h}_{t-1}\right) \tag{2.25}$$

$$\mathbf{h}_t = \mathbf{u}_t \odot \mathbf{h}_{t-1} + \left(1 - \mathbf{u}_t\right) \odot \widetilde{\mathbf{h}}_t \tag{2.26}$$

Temporal Convolutional Networks

Temporal Convolutional Networks (TCN) show better results for time series modeling than RNN [43]. Those networks are Convolutional Neural Networks (CNN) applied to data with temporal correlation, they are also named 1D Convolutional Networks, because they use convolutions along only one dimension, although its inputs and outputs can have multiple dimensions. A comprehensive tutorial of CNNs can be found in [44].

Recent best practices in TCN include the use of causal and dilated convolutions and residual connections. Dilated convolutions are layers that improve the receptive field of the output by skipping input values at a specific interval, thus applying a filter over a larger area than it would usually do. A receptive field from a standard causal convolution would be given by $R = L + K - 1$, where R is the receptive field, L is the number of layers and K is the kernel size.

In contrast, stacked dilated convolutions, where each layer has the dilation doubled starting from 1, have a receptive field given by $2^{L+1} - 1$ [45] for $K = 2$. Dilation is related to strides and pooling, as they skip some time steps to compute the convolution, but dilated convolutions return a same-sized window as output, while stride and pooling are used to downsample windows. Dilated kernels are multiplied by a mask with zeros, which defines data points to be skipped, in order to return a same-sized window.

The name causal convolution comes from layers that use causal padding, where output at time t does not depend on input at time $t + 1$, in other words, the layer only uses previous time-steps to make its prediction $p(x_{t+1}|x_1, \ldots, x_t)$. Causal padding is implemented by left padding $D \cdot (K - 1)$ zeros to the layer input, where D is the dilation rate. There are residual and skip connections which speed up training convergence and enable deep models by connecting spatially distant layers, and improving gradient flow through the network, thus disabling gradient vanishing problems.

BAI *et al.* [43] list some advantages in comparison with RNNs models. Among the advantages, we highlight: (1) parallelism – an RNN must wait for the previous output, while a TCN can simultaneously process all entries of an input vector, (2) configurable receptive field size – it is possible to configure a desirable receptive field during modeling phase, while in RNNs it is not, and (3) less memory requirement – as the number of cells, and thus gates of an LSTM, increases, the use of memory also increases, while in TCNs the weights are shared inside layers and, in practice, use less memory than RNNs. Moreover, [43] shows that TCN retains information for longer sequences than LSTM and GRU, even though RNNs can theoretically retain information for an indefinite amount of time. Due to those advantages, we use WaveNILM – a TCN model – to compare it with FHMM in an industrial load disagreggation setting.

ANN in NILM

KELLY e KNOTTENBELT [14] use LSTM in their work with NILM and compare it with CNN models. They claim that LSTM does not perform well for multi-state appliances, probably because of how events can be separated by more than 1000 time steps, and LSTM yields overall error greater than their CNN implementations. DO NASCIMENTO [20] trains an LSTM model, alongside GRU [42] and CNN models. In his work, LSTM is outperformed by all other models, and GRU yields the best results.

2.3 Factorial Hidden Markov Models

Another widely used method in time series processing is called HMM. It is defined as a non-observable stochastic process inside a stochastic process [46]. All states in an HMM are hidden and are a probabilistic function of an output, which is observable. In other words, for an HMM, we observe a stochastic process $\{Y_t\}_{t\geq 0}$ that is linked to a hidden Markov Chain $\{S_t\}_{t\geq 0}$, where $t \in \mathbb{Z}^+$. A first-order Markov chain is a sequence of states where the state at time t only depends on the state at time $t-1$.

When applied to NILM, every appliance is modeled as an HMM and its internal states are modeled as Markov Chains. PARSON *et al.* [15] model several household appliances into HMMs to solve NILM, and assume that every step change in the whole-building aggregate power is an observation from the sequence of an appliance changing state. This method does not consider multiple simultaneous step changes from different individual appliances. Figure 2.5 shows an HMM representation.

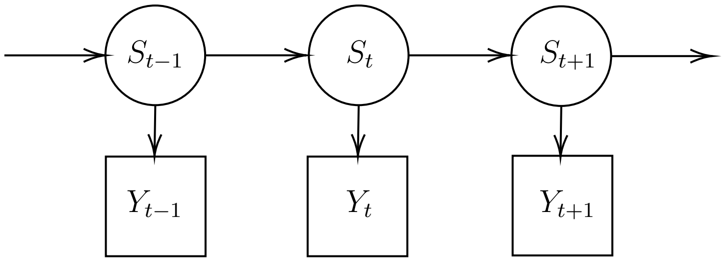


Figure 2.5: HMM representing a single device. Circumferences are hidden states while squares are observable outputs.

As the number of appliances in a building increases, the probability of multiple simultaneous step changes increases, and in certain circumstances they are expected, for example a factory that turns on all machines at the same time of the day. Factorial Hidden Markov Models (Modelos Ocultos Fatoriais de Markov, FHMM) (Fig. 2.6) in NILM [16] address this issue by considering every observed output as an additive function of different hidden step changes of appliances. On an FHMM [47] each appliance is modeled as an independent HMM and the output of each model is added to generate the observable output. Each appliance, therefore, runs independently and simultaneously.

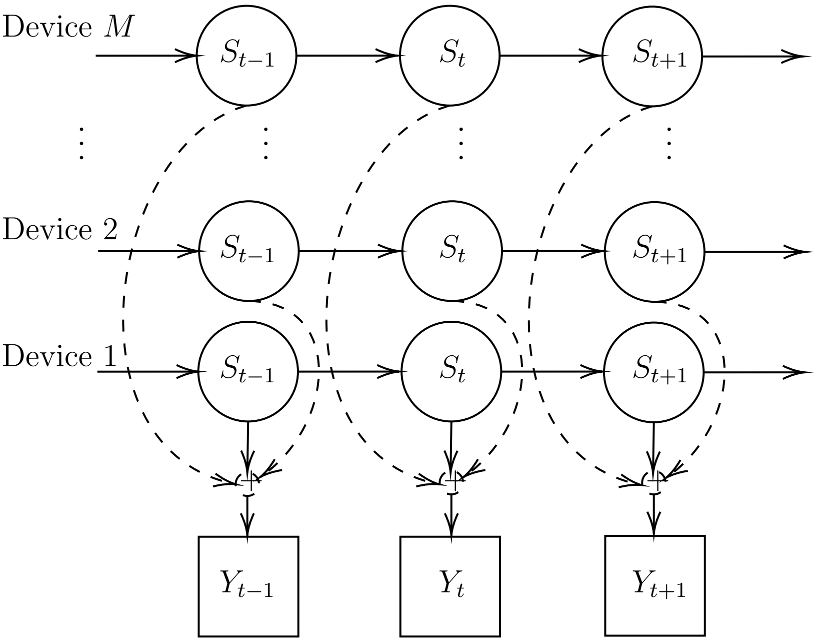


Figure 2.6: Representation of an FHMM with M devices. The output Y_t is the sum of all inner HMM outputs.

Chapter 3

Data set

In this chapter, we present the data set IMDELD [1], which is used to train, validate and test the models implemented for this work. Several NILM data sets [48–54] are available publicly. They are usually time series of electrical features of appliances and circuits collected with the principal purpose of training and testing NILM algorithms. As NILM is an umbrella term for different approaches of appliance load disaggregation from buildings, all those data sets have different settings such as measured features, sampling rate, and how many buildings they refer to. Table 3.1 shows these differences. All of them have different duration of data collection, ranging from seconds to years.

Table 3.1: Some publicly available NILM data sets and their differences. LF stands for low frequency and HF for high frequency, while V is voltage, I is current, P is active power, Q is reactive power, S is apparent power, E is active energy and THD is total harmonic distortion.

Data set		Features	Sampling rate	Buildings	Appliances
IMDELD	[1]	V, I, P, Q, S, E	1 Hz	1	8
REDD - LF	[48]	P	1 Hz	6	24
REDD - HF	[48]	V, I	16.500 Hz	2	0
UK-Dale - HF	[49]	P	16.000 Hz	3	0
UK-Dale - LF	[49]	P	0,1667 Hz	5	up to 52
GREEND	[50]	P	1 Hz	8	9
BLUED	[51]	V, I	12.000 Hz	1	43
WHITED	[52]	I	44.000 Hz	1	46
COOLL	[53]	V, I	100.000 Hz	1	42
Dataport	[55]	V, P, S, THD	1 Hz	+1000	+70

WHITED [52], and COOLL [53] are data sets of appliances and do not have whole-building measurements. Out of all the cited [48–54] data sets, only WHITED has some industry appliances including a treadmill, two soldering irons, a sewing machine, a jigsaw, and a bench grinder. However, those are all from light industry environments and do not count as heavy machinery. The rest is focused on residential appliances like fridges, air conditioners, electric ovens, and computers. As part of the project, IMDELD was collected specifically for this work. The author found no other publicly available data set with industry machinery.

3.1 Factory

As stated above, IMDELD is a public data set; therefore, we have to take some notes on privacy concerns. The factory accepted having their electrical signature shared for the only purpose of advancing research in NILM or correlated areas. The company decided to share this data anonymously. In this way, we will not give details that could violate their right to stay unidentified.

Eleven energy meters collected the data in a poultry feed factory in the state of Minas Gerais, in Brazil. Its process is the same all year round, working from Mondays through Fridays, and occasionally on Saturdays, which happens when the set monthly target is not met. It has three daily rotating shift work hours, from 22:00 to 17:00, because electricity prices are higher from 17:00 to 22:00 and the factory closes.

The factory produces poultry ration with corn or soybeans and added nutrients. The food is milled and mixed to create a homogeneous paste before going to a machine that creates pellets out of those. During its work hours, the factory works at full-scale producing as many pellets as possible.

All of that makes the electrical consumption of the factory consistent and generally independent of the day, even during holidays at workdays it is at least partially open. The local energy provider sells medium voltage (13.4 kV) energy to the factory. This energy supply is connected to a local substation that transforms medium voltage to low voltage (380 V) and connects it to four different Low Voltage Distribution Board (LVDB).

Each LVDB provides energy to a specific area inside the factory: one (LVDB-1) for lights and administration-related appliances, the second (LVDB-2) for pelletizing-related machinery, the third (LVDB-3) for milling-related machinery, and the last one (LVDB-4) for general production-related machinery. Figure 3.1 shows this relationship.

Sub-circuits under an LVDB are connected via a four-wire three-phase system (one for ground and the others for phases A, B, and C). All motors connected to LVDB-2 and LVDB-3 were assembled from 1998 to 2008 and are three-phase Direct Current (DC) motors. The motors are responsible for the most considerable amount of energy consumption in each of these two sub-circuits. The three-phase system is well-balanced, creating at most a 5% power difference between each phase.

There are two types of appliances under the LVDB-2 circuit: pelletizers and

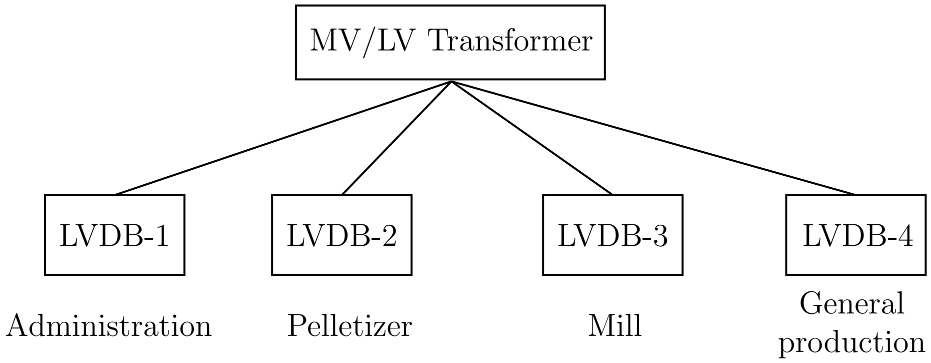


Figure 3.1: Factory circuits. MV/LV Transformer stands for Medium-Voltage/Low-Voltage Transformer. MV/LV Transformer is the main site circuit, located at the factory substation.

exhaust fans. Under the LVDB-3 there are milling machines and exhaust fans. The general production-related LVDB supplies energy mainly to the belt conveyor system and also to the maintenance related-machinery. The second and third LVDBs account for up to 80% of the total energy consumption of the factory, as they supply the largest machines.

It is important to note that DC motors inject much inductive reactive energy into the grid. To prevent fines from the energy provider, the factory has a passive capacitor bank inside the substation. It is said to be passive because it is always connected to the grid, thus not disconnected when the motors are turned off. This bank sets a capacitive constant in reactive power at all time.

3.2 Hardware

The hardware used to measure every appliance and circuit in this data set is a GASM-B [56]. It is a three-phase meter that connects via Wi-Fi to the internet and sends the data collected to GreenAnt's database. It can save up to 30 days of measurements if it has no internet connection. This was very useful on the industrial site, as the connection was not always available due to remoteness of the region. A picture of a GASM-B can be seen in Fig. 3.2.

GASM-B uses up to three Current Transformer (CT)s to collect current data from electrical conductors and a four-wired connector to collect the electric potential difference from different phases. The meter samples current and voltage at 8 kSamples/sec to compute active power, reactive power, apparent power, active energy, and reactive energy. This data is then downsampled to 1 Hz and stored as a package of RMS voltage, RMS current, active power, reactive power, apparent power, active energy, and reactive energy in the internal memory. Every ten seconds, it sends a package of measurements to the Wi-Fi connection. As internet is unstable in the region, the data was sent directly to an on-site server and later uploaded to our remote server when the internet was available.

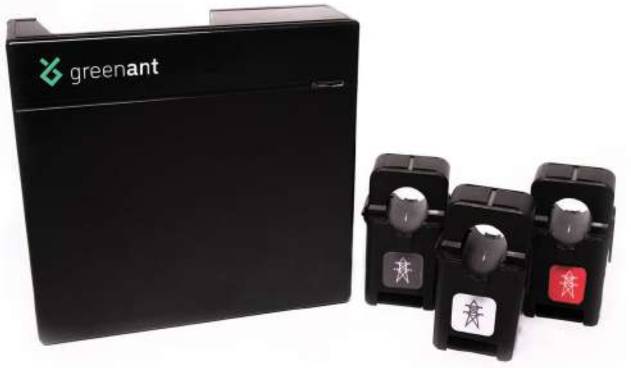


Figure 3.2: Picture of a GASM-B and three CTs.

3.3 Installation

This project had access to eleven meters for data acquisition; hence this was the limit of appliances and circuits that could be measured. In order to analyze the most critical circuits, it was decided to install meters in LVDB-2, LVDB-3 and before the MV/LV. This decision was chosen due to the importance of both sub-circuits to the factory. Figure 3.3 shows that one out of LVDB-1 or LVDB-4 can consume more mean active power than the LVDB-3.

Installing the meter before the MV/LV Transformer means that the data collected directly refers to the supply from the energy provider, at 13.8 kV voltage. As the main input for the disaggregation models, the meter before the MV/LV Transformer is considered the site meter. Figure 3.4 shows an example of power consumption measured from the site meter.

Besides the site meter, other two meters were installed in the circuits LVDB-2 and LVDB-3. Those can be used as both input and output for the disaggregation models. In NILM, they can be seen as outputs if it is interesting to know the energy and power consumption of a whole factory sector. All machines under LVDB-2 and LVDB-3 work at 380 V and 60 Hz, and not all machines were chosen for measurement, only those asked by the factory.

This leaves eight meters to be allocated by appliances. Under LVDB-2, the appliances measured are PI, PII, EFI, EFII, DPI, and DPII. All of them usually turn on together, although not at the same time as a measurement to prevent fines on high power consumption. Double-pole contactors are important contactors inside the sub-circuit, and the factory asked GreenAnt to measure them. Current

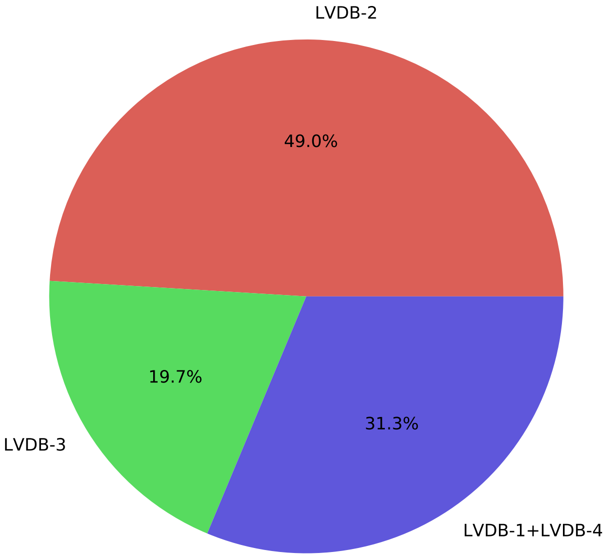


Figure 3.3: Percentage of average active power demand over the measurement time interval of LVDBs 2 and 3 in comparison with other LVDBs under the MV/LV Transformer.

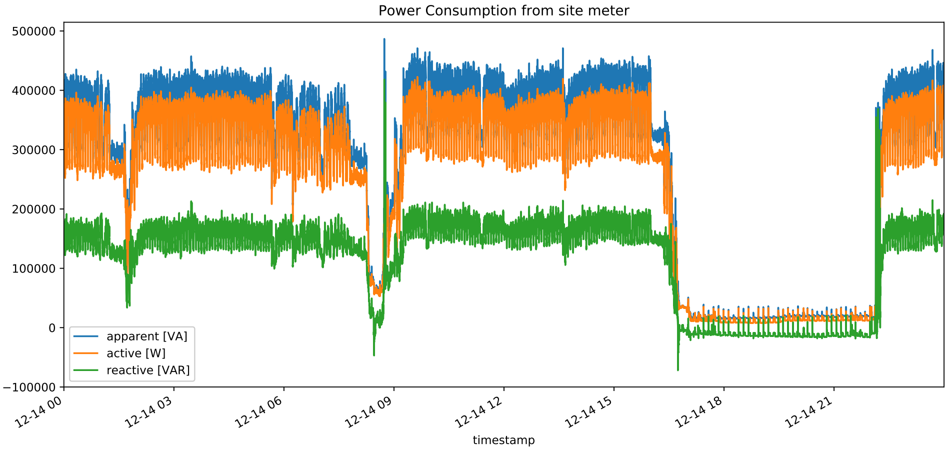


Figure 3.4: Power consumption measured from the site meter (meter before the MV/LV Transformer). Timestamps are in MM-DD HH format (month-day hour). Negative reactive power means capacitive reactive power, while positive means inductive. (Reprinted with permission from MARTINS *et al.* [23])

consumption on each pelletizer can reach up to 1000 A, and an electrical engineer had to install the CT on the pelletizer's energy supply, as seen in Fig. 3.5.

Figure 3.5: Installation of a current transformer by an electrical engineer on an appliance energy supply.

The last two meters were set at the last 12 days of measurement under the LVDB-3, as they were only available for installation during those days. They measured both MI and MII. All eight appliances measured are used for the ground-truth analysis of the disaggregation models; all measured circuits and appliances by meters can be seen in Fig. 3.6. Samples were collected from December 11$^{\text{th}}$ 2017 at 18:43:52 UTC until April 1$^{\text{st}}$ 2018 at 21:33:17 UTC, or roughly 111 days. This interval means that MI and MII only have measurements from about the last 10.81% of the total measurement time.

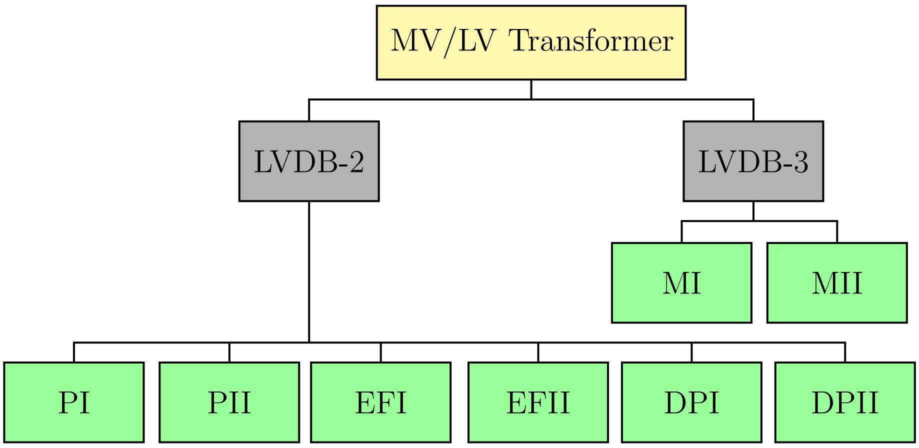


Figure 3.6: Diagram showing measured circuits and appliances. Yellow block stands for site-meter, gray blocks stand for LVDBs, and green blocks stand for appliances. Only the site-meter and appliances were set for load disaggregation.

3.4 Details

Some further details can be examined with the collected data. Figures 3.7, 3.8, 3.9, 3.11, 3.12 were generated with NILMTK [8] python module.

Figure 3.7 shows a two-day time frame from March 31st 2018 to April 2nd 2018, of active power demand from all meters in the factory. Although it was previously written that the factory does not work from 17h00 to 22h00, March 31st was a Saturday, so the factory probably had to work extra hours to achieve a production target. This example shows an atypical day during the week, in which the factory worked on full scale from 07h00 until 04h00. Here only the active power for each meter is shown because all of the time-series features have high correlation with the active power, as all of them are created from the sampled current and voltage waveforms.

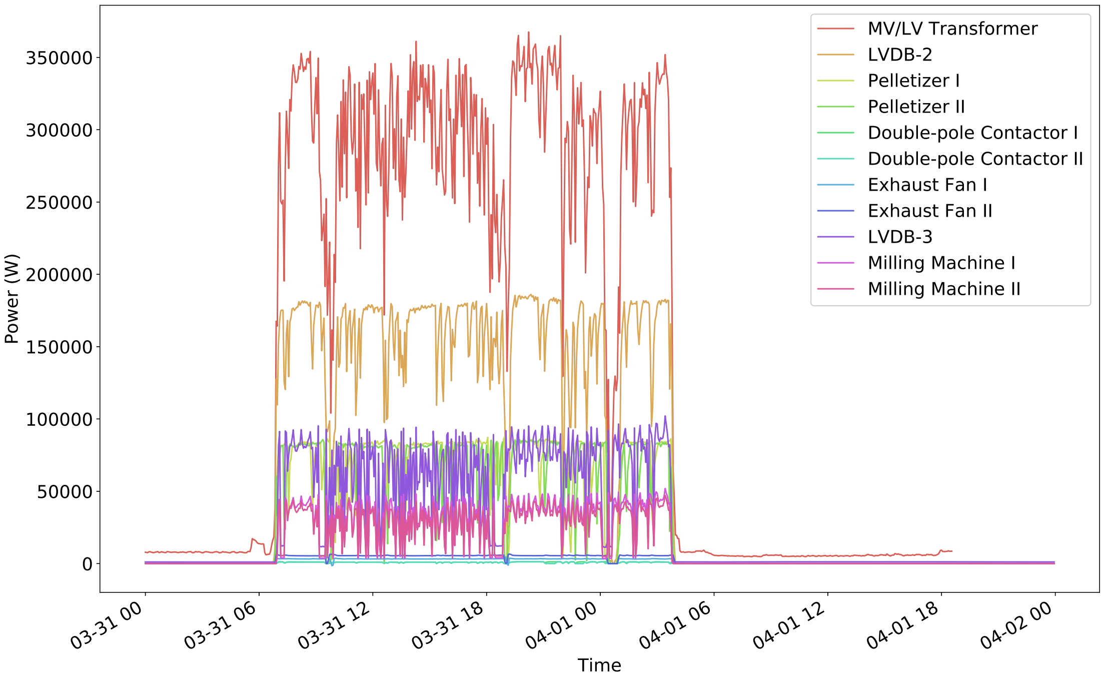


Figure 3.7: Active power from all meters from 2018-03-31 to 2018-04-02. Timestamp is defined in month-day hour format.

The time when each appliance and circuit demands active power from the network is shown in Fig. 3.8. As the meter responsible for collecting data from the MV/LV transformer had to be installed on the factory substation and at a 13.8 kV electric tension, it demanded greater care in comparison with other circuits and appliances to be installed. This attention delayed its installation until December 11th 2017. On the other hand, the data set has measurements from LVDB-2, PI, PII, DPI, DPII, EFI, and EFII since October 30th 2017, as it was possible to install the meters on the first factory visit. The last three meters were installed on February

19th 2018 to collect LVDB-3, MI, and MII.

Looking when MV/LV transformer had gaps in this chart displays when data were lost due to internet connectivity or meter malfunction. Gaps in sub-meters (appliances or sub-circuits) can also correspond to work-free days instead of internet or meter malfunction. This chart is essential to know where are time frames without data loss.

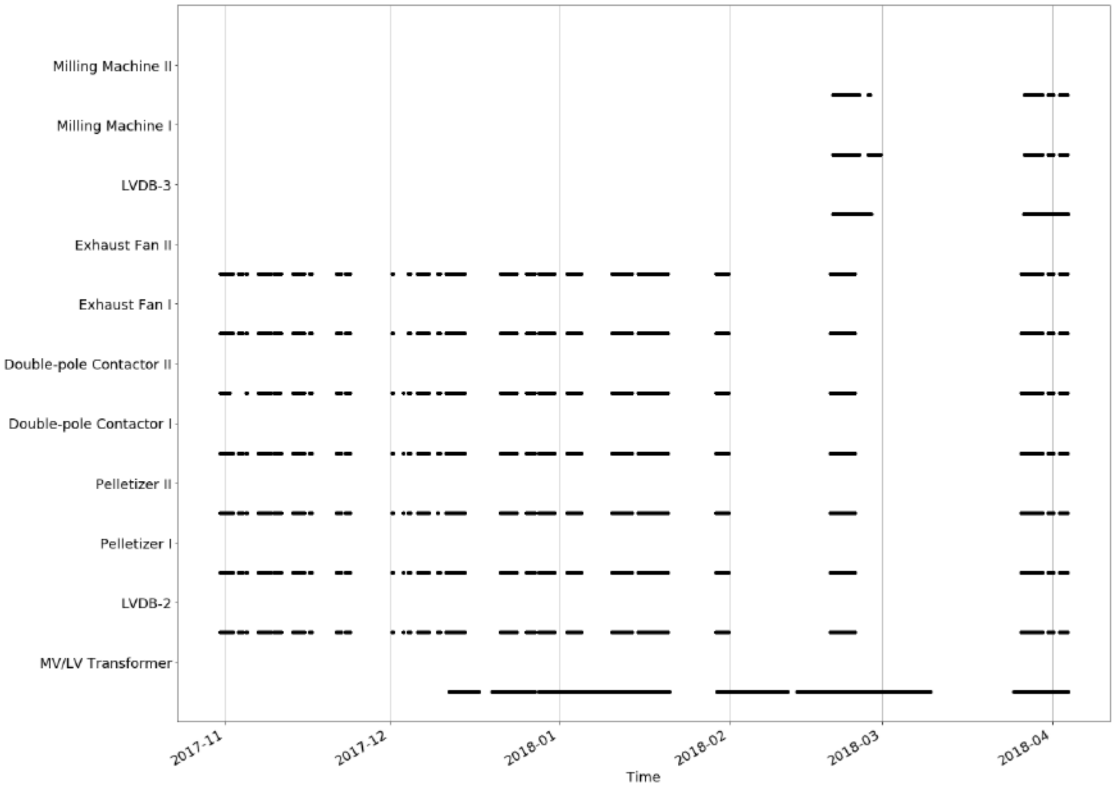
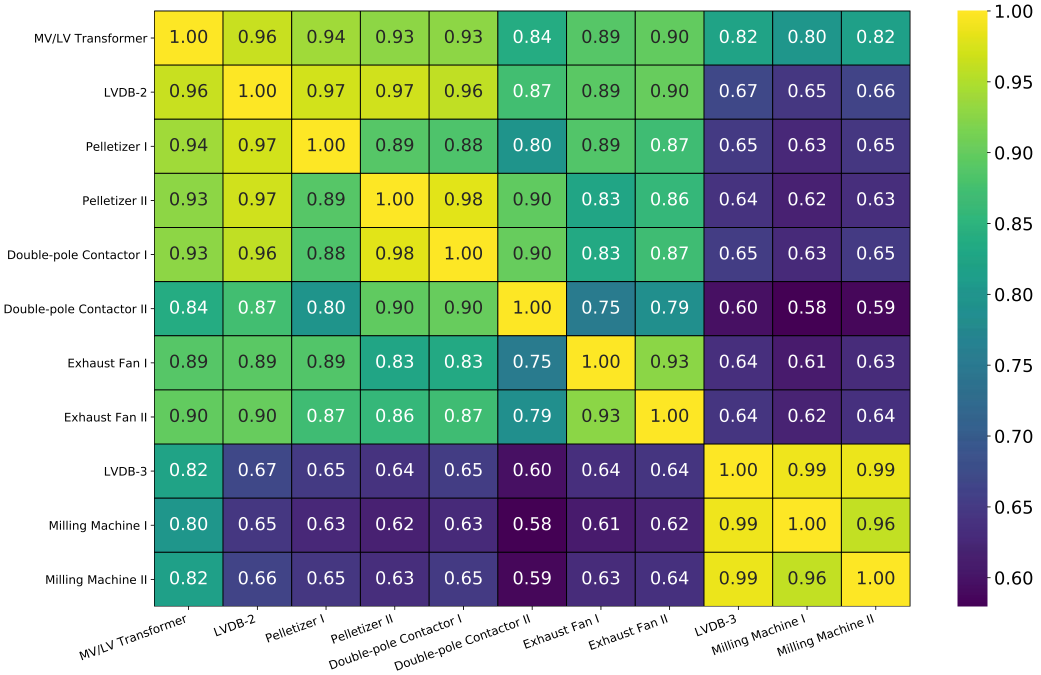


Figure 3.8: Overview of time intervals in which each circuit or appliance consumes energy. Timestamp is defined in year-month format.

Figure 3.9 shows the active power pairwise Pearson correlation between each measured circuit and appliance from February 23rd 2018 until the last measurement. It is interesting to notice how appliances inside the same circuit have a high correlation. This correlation happens because appliances connected in the same circuit are usually working at the same time, and the circuit itself is the sum of all appliances. Correlations between two sub-measurements of different circuits (one from LVDB-2 and another from LVDB-3) are not as high, lying on a range from 0.58 to 0.67. This information is interesting to show us how those two LVDBs are isolated from each other, WaveNILM was first trained in LVDB-2 and later fine-tuned in LVDB-3. DPII is the least correlated machine with all other appliances, this impacts the results of both WaveNILM and FHMM in load disaggregation, as it does not follow the same signature as MV/LV Transformer a hundred percentage of the time. Pelletizers have high influence in the energy consumption of the whole factory, as seen in Fig. 3.11, thus it is expected that they would have the highest correlation with the site-meter, but interestingly enough, DPI has a high correlation as well, showing that it probably has an electrical signature similar to pelletizers.



	MV/LV Transformer	LVDB-2	Pelletizer I	Pelletizer II	Double-pole Contactor I	Double-pole Contactor II	Exhaust Fan I	Exhaust Fan II	LVDB-3	Milling Machine I	Milling Machine II
MV/LV Transformer	1.00	0.96	0.94	0.93	0.93	0.84	0.89	0.90	0.82	0.80	0.82
LVDB-2	0.96	1.00	0.97	0.97	0.96	0.87	0.89	0.90	0.67	0.65	0.66
Pelletizer I	0.94	0.97	1.00	0.89	0.88	0.80	0.89	0.87	0.65	0.63	0.65
Pelletizer II	0.93	0.97	0.89	1.00	0.98	0.90	0.83	0.86	0.64	0.62	0.63
Double-pole Contactor I	0.93	0.96	0.88	0.98	1.00	0.90	0.83	0.87	0.65	0.63	0.65
Double-pole Contactor II	0.84	0.87	0.80	0.90	0.90	1.00	0.75	0.79	0.60	0.58	0.59
Exhaust Fan I	0.89	0.89	0.89	0.83	0.83	0.75	1.00	0.93	0.64	0.61	0.63
Exhaust Fan II	0.90	0.90	0.87	0.86	0.87	0.79	0.93	1.00	0.64	0.62	0.64
LVDB-3	0.82	0.67	0.65	0.64	0.65	0.60	0.64	0.64	1.00	0.99	0.99
Milling Machine I	0.80	0.65	0.63	0.62	0.63	0.58	0.61	0.62	0.99	1.00	0.96
Milling Machine II	0.82	0.66	0.65	0.63	0.65	0.59	0.63	0.64	0.99	0.96	1.00

Figure 3.9: Pairwise active power correlation between each measured circuit and appliance. This heatmap represents only the days when all appliances had measures, more specifically from 2018-02-23 until the end.

Figure 3.10 shows the pairwise Pearson correlation between each measured feature in the MV/LV transformer. As expected, every feature is highly correlated with each other besides the voltage. The voltage has a low correlation with the other features because it stays highly stable while current and power fluctuate. It also has a negative Pearson correlation index, and this can probably be explained by the high current consumption, thus changing cable resistance and causing voltage to decrease.

Considering only appliances, the pie chart in Fig. 3.11 displays the percentage, of total appliance energy consumption measured, that each appliance consumes. This percentage shows how pelletizers are the biggest consumers, while the other appliances consume only 9.8% of all energy consumption. This trend can be further viewed in Tab. 3.2.

Table 3.2 shows how each appliance works when they are turned on. The PI, PII, MI, MII power consumption is two orders of magnitude above the power consumption of EFI, EFII, DPI and DPII. The double-pole contactors are the smallest consumers, followed by the exhaust fans, then milling machines, and finally the pelletizers. Each type of appliance has its distinctive reactive power consumption,

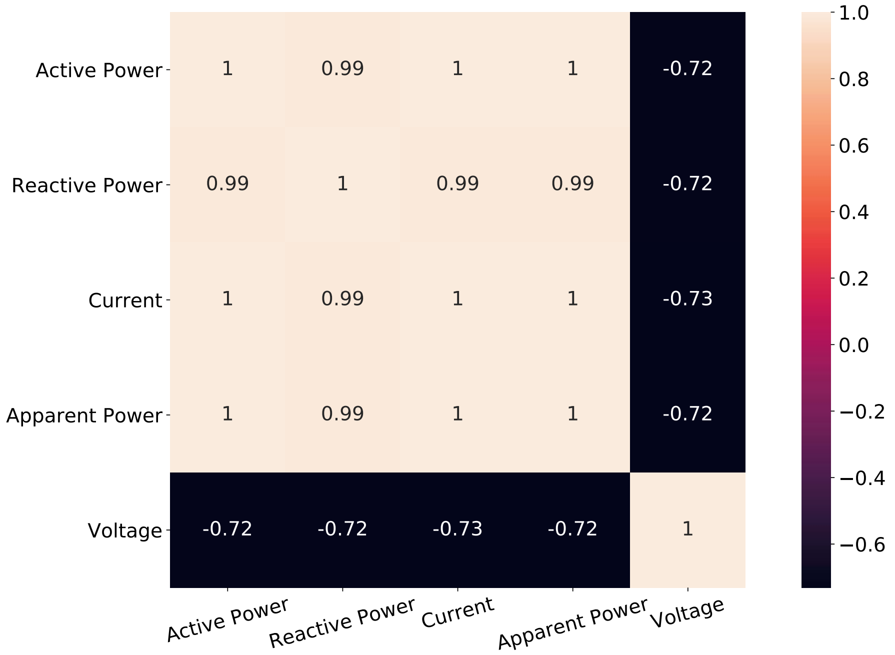


Figure 3.10: Pairwise correlation between each feature of the MV/LV transformer circuit.

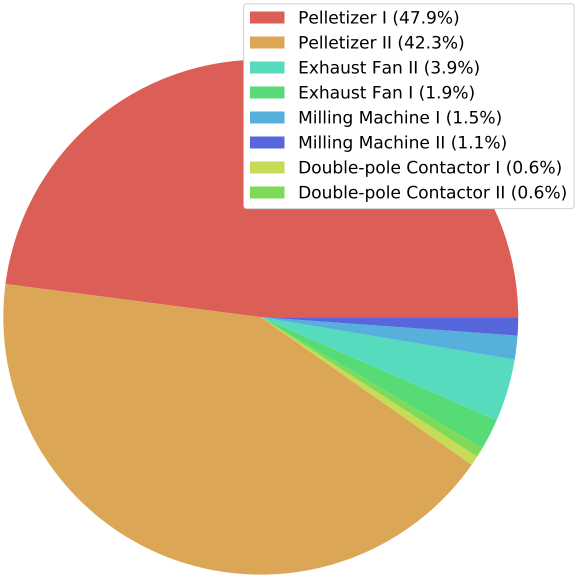


Figure 3.11: Percentage of overall energy consumption corresponding to each appliance.

and also that voltage is somewhat stable across each device. Histograms of each appliance during ON time can be seen in Appendix A.

Figure 3.12 shows the autocorrelation $\mathbf{R_{XX}}(\tau) = \mathbf{E}\left[\mathbf{X}_t \mathbf{X}_{t+\tau}\right]$ measured from the MV/LV Transformer active power demand. As τ increases, $\mathbf{R_{XX}}$ decreases. After 263 lag points (τ), $\mathbf{R_{XX}}$ is below 0.5. Each lag point is a second, so it takes 4 minutes and 23 seconds to set signal autocorrelation below 0.5. This analysis is important to set a window size for training convolutional neural networks and is explained in the last part of Sec. 4.2.

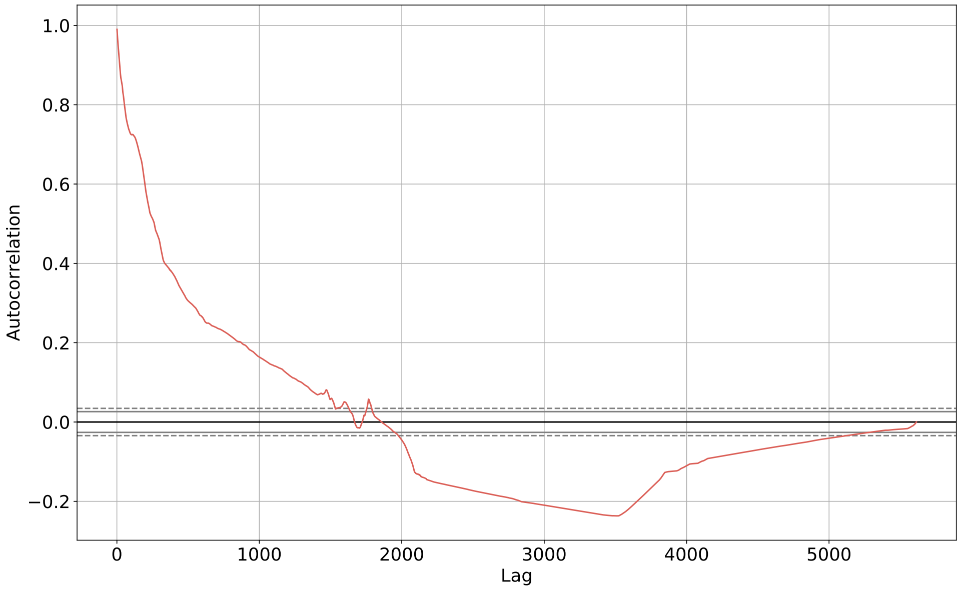


Figure 3.12: MV/LV Transformer autocorrelation versus data points lag.

Table 3.2: Description of mean and standard deviation for measured features for all appliances during steady-state.

Appliance	Active Power [W]	Reactive Power [VAr]	Apparent Power [VA]	Current [A]	Voltage [V]
Pelletizer I	$(8.1 \pm 1.1) \times 10^4$	$(52.7 \pm 4.9) \times 10^3$	$(9.7 \pm 1.1) \times 10^4$	$(4.5 \pm 0.5) \times 10^2$	216.5 ± 2.6
Pelletizer II	$(7.8 \pm 1.2) \times 10^4$	$(52.1 \pm 5.7) \times 10^3$	$(9.4 \pm 1.2) \times 10^4$	$(4.4 \pm 0.6) \times 10^2$	216.2 ± 2.3
Double-pole contactor I	$(11.0 \pm 2.3) \times 10^2$	$(34.0 \pm 1.3) \times 10^2$	$(35.9 \pm 1.2) \times 10^2$	$(16.4 \pm 0.4) \times 10^0$	218.2 ± 2.8
Double-pole contactor II	$(11.1 \pm 2.3) \times 10^2$	$(33.7 \pm 1.2) \times 10^2$	$(35.6 \pm 1.0) \times 10^2$	$(16.4 \pm 0.3) \times 10^0$	216.1 ± 2.8
Exhaust Fan I	$(32.8 \pm 2.9) \times 10^2$	$(11.2 \pm 2.5) \times 10^2$	$(34.8 \pm 2.9) \times 10^2$	$(15.9 \pm 1.2) \times 10^0$	218.6 ± 3.2
Exhaust Fan II	$(56.1 \pm 2.9) \times 10^2$	$(6.9 \pm 2.6) \times 10^2$	$(88.6 \pm 3.3) \times 10^2$	$(41.1 \pm 1.1) \times 10^0$	215.6 ± 3.1
Milling Machine I	$(4.4 \pm 1.4) \times 10^4$	$(34.3 \pm 5.0) \times 10^3$	$(5.6 \pm 1.3) \times 10^4$	$(2.8 \pm 0.7) \times 10^2$	205.8 ± 4.8
Milling Machine II	$(3.1 \pm 1.9) \times 10^4$	$(22.9 \pm 6.1) \times 10^3$	$(4.0 \pm 1.8) \times 10^4$	$(1.9 \pm 0.9) \times 10^2$	209.7 ± 6.9

Chapter 4

Method

This chapter describes the method to disaggregate industrial heavy machinery loads from total factory energy consumption studied in this thesis. First, in Sec. 4.1, we talk about the frameworks used, then Sec. 4.2 shows how the data set described in Chap. 3 is applied throughout the models. In Sec. 4.3, the implementation of WaveNILM is given in details.

4.1 Frameworks

FHMM and WaveNILM implementations use different frameworks. The python module NILMTK[1] [8] has one FHMM algorithm implementation, and it was chosen as the FHMM used in the code, this would guarantee that it is the same one used in other research efforts like [16]. WaveNILM was implemented in Keras [7] based on WaveNet [57].

NILMTK is a python module designed to aid comparing disaggregation algorithms, and it is useful because it has a collection of algorithms like FHMM and standard accuracy metrics implemented. It also has documentation that helps writing new disaggregation algorithms, metrics and data set importers. It uses NILM Metadata [58], which defines classes and relationships between classes for NILM data sets. The IMDELD data set was transformed to be compatible with NILMTK using NILM Metadata schema. The original data set in CSV format and the transformed data set in HDF5 format can be downloaded at [1].

4.2 Training, validation and test sets

For WaveNILM and FHMM, the data set was divided into three different sets: a training set, a validation set and a test set. The validation set was used to choose

[1] https://github.com/nilmtk/nilmtk/

the best model. The test set was fixed before the training and was only shown to the models after validation; the results shown in the present text are generated with the test set.

FHMM models every appliance at the same iteration and, to make sure that an FHMM is trained and validated with at least 85% of the MI and MII data, the test set size is fixed at the last 15% of LVDB-3 data. This means that the test set starts at the timestamp 2018-04-01 16:17:21-03:00 and includes data until the last measurement. Table 4.1 defines each set for FHMM training. The test set used is shown in Fig. 4.1. The FHMM implementation in NILMTK always receives active power as input and output; this is in accordance with other works that used this method for load disaggregation.

Table 4.1: Training, validating and test sets for FHMM models. All timestamps are in UTC.

Set	Start	End	Pct of data
Training	2017-12-11 16:43:52	2018-03-30 17:16:34	92.78%
Validating	2018-03-30 17:16:35	2018-04-01 13:17:20	3.96%
Test	2018-04-01 13:17:21	2018-04-03 15:48:47	3.26%

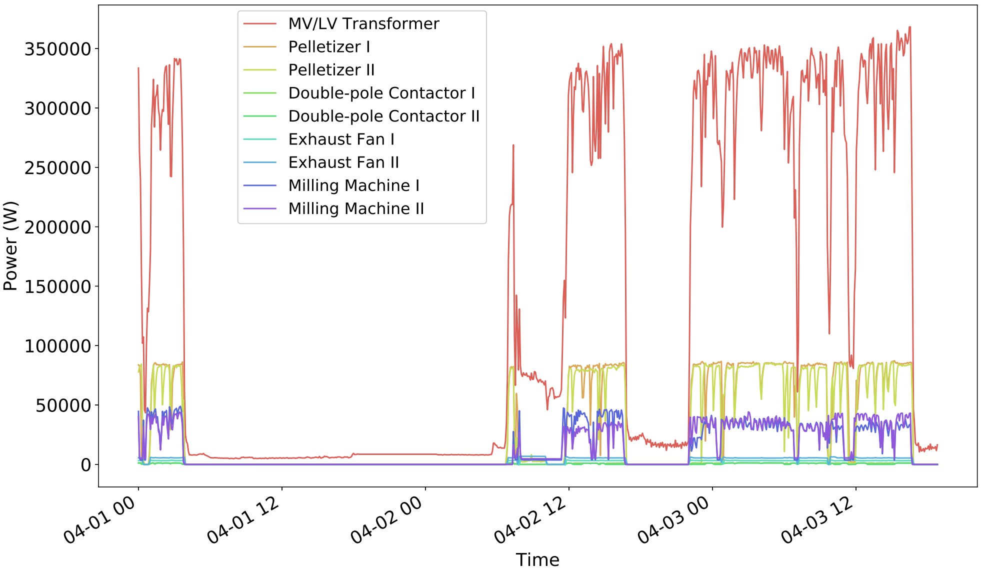


Figure 4.1: Test set used for the FHMM and WaveNILM results.

Instead of trying to model every appliance at the same time, WaveNILM learns only one device per model. In this manner, there are eight groups of training, validating and testing sets; each group is used to generate a different WaveNILM model. So there are eight different specialized deep learning models, one for each appliance. Each group consists of features that share the same timestamp for the MV/LV Transformer and the desired appliance. The neural networks use current and voltage from the MV/LV Transformer as inputs, while the active power from the appliance is used as a target. As shown in Fig. 3.10, active power and current are highly correlated, while voltage is mildly negative correlated with the other two. RMS current and RMS voltage are chosen as input features as they are readily available and they can create the RMS apparent power, by multiplying both, thus yielding a third indirect feature. Active power is decided to be the output as it is the most important feature according to the factory.

The test set for all appliances is chosen to be the same as the one used in the FHMM model, while the rest of the data is used for training and validation. PI, PII, DPI, DPII, EFI, and EFII use the interval from 2017-12-11 16:43:52 to 2018-04-01 13:17:16 for training and validation, while MI and MII use the interval from 2018-02-19 19:52:28 to 2018-04-01 13:17:16.

After selecting a group, the data selected for training and validation is divided into seven folds. WaveNILM model training is thus repeated seven times. For each iteration, a different fold is chosen as a validation set, and the other six are merged as the training set. This folding was done in order to provide information on how the model changed according to the training set. The validation set is used to find the best-trained model.

Means (μ) and standard deviations (σ) are computed from the training set for each feature and then used in the normalization of the model input, as show in Eq. (4.1). Voltage is normalized with $\mu_V = 6743$ V and $\sigma_V = 50$ V, while current is normalized with $\mu_I = 30$ A and $\sigma_I = 24$ A. Targets are not normalized in this setting.

$$z = \frac{x - \mu}{\sigma} \tag{4.1}$$

WaveNILM models evaluate windows of 1024 normalized samples with two features – current and voltage – and output a 1024 window of active power from the modeled appliance. This widows size was decided based on Fig. 3.12, as in a window of 1024 samples there is still meaningful correlation in between timestamps. If the total number of seconds of a set is not a multiple of 1024, the last seconds are removed until it became one such multiple. PI, PII, DPI, DPII, EFI, and EFII are trained with 2677 samples and evaluated with 446 window samples. Five hundred forty-seven (547) windows are available for MI and MII training and validation. All sets are shuffled before training and validation. The test set has 121 windows used in the final evaluation of the models.

4.3 WaveNILM implementation

Another approach to the NILM problem is to use deep neural networks, as shown in residential data by KELLY e KNOTTENBELT [14], ZHANG *et al.* [18] and MORGAN [21]. WaveNILM is a deep learning model based on Google DeepMind WaveNet [57]. WaveNet was developed to generate raw audio waveform, and is used on several sub-fields inside audio generation such as text-to-speech, music generation, voice conversion and source separation. Load disaggregation can be seen as an appliance-load generation based on site meter data, therefore it relates to source separation using an audio waveform generator model like WaveNet.

WaveNILM, like WaveNet, is a deep learning model for time series, thus it uses temporal convolutions (also called 1D convolutions). It uses three types of temporal convolutions: 1 × 1 convolutions (or 1D convolutions with kernel size 1), causal convolutions, and dilated causal convolutions. 1D convolutions with kernel size 1 are used to permute channels and set outputs with the desired channel size. Figure 4.2 shows the general flowchart of the model, and Fig. 4.3 details a WaveNet block. If we count that B blocks are used on WaveNet, then the receptive field equation is given by Eq. (4.2).

$$R = B \cdot 2^{L+1} - B + 1 \tag{4.2}$$

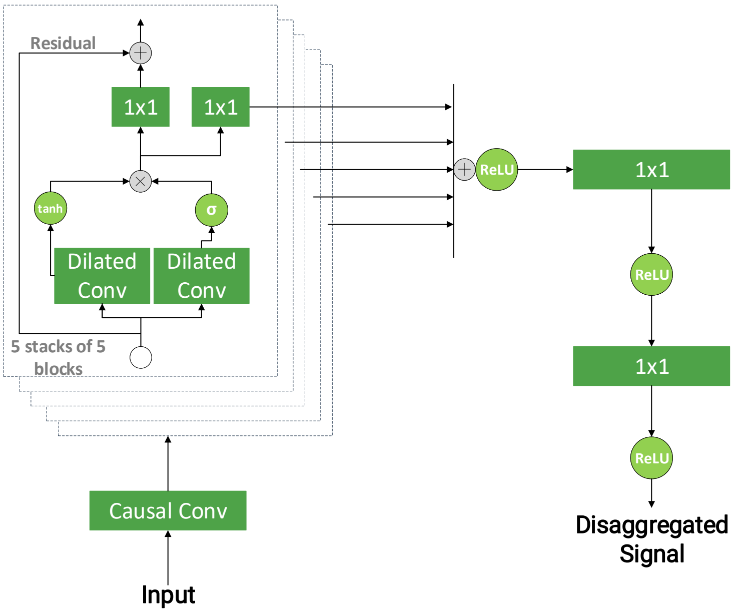


Figure 4.2: WaveNILM model based on DeepMind Wavenet [57]. Reprinted with some modifications and permission from [23]. A σ represents a sigmoid function.

Eight deep models are trained in the present work. Each model is trained to disaggregate one specific machine inside the factory. One model is pre-trained on

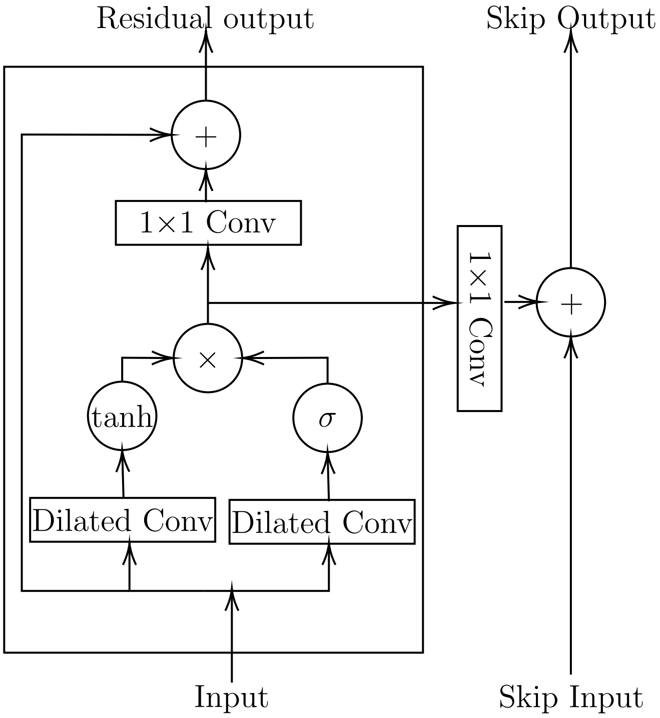


Figure 4.3: A detailed version of the implemented WaveNet block, also used in WaveNILM.

the PI machine for a hundred epochs. It is then fine-tuned for all seven remaining appliances, thus creating the other seven models, for twelve epochs. This pre-trained model increases model convergence speed, with some appliances leading to convergence at the sixth epoch. PI is chosen as the pre-training machine for other appliances, because it has the most extensive data set and the most significant impact on the factory energy consumption. Other machines were tested for pre-training input, but besides MI and MII, which have fewer examples and worse results, they yield similar validation loss. All models are trained with Adam optimization algorithm and MAE loss function. The log file created for this setting with further information can be seen in Appendix C. Appendix B shows charts with all training and validation losses.

General hyper-parameters used for all models can be seen in Tab. 4.2. Each causal convolution, including the ones with dilated padding, uses 32 kernels with 2 weights, 1×1 convolutions use 32 kernels of size 1 followed by linear activation functions, and the output 1×1 convolution is a kernel of size 1 with ReLU activation. The models have five stacks of four dilation depth layers, meaning that each stack has five WaveNet blocks with five stacked layers with dilation $0, 1, 2, 4,$ and 8, respectively, which yields a 156-seconds receptive field for each time step at the output, according to Eq. (4.2). A 156-seconds window has $0.68 \leq \mathbf{R_{XX}}(\tau) \leq 1.00$, according to Fig. 3.12, between each data point and a lag τ.

Table 4.2: WaveNILM hyper-parameters.

Hyper-parameter	Value
Number of stacks	5
Number of filters	32
Dilation depth	4
Window size	1024
Number of epochs	100
Loss function	MAE
Optimization	Adam

4.4 Metrics

After choosing each model from a fold based on MAE computed at the validation, the trained models are validated according to four different metrics computed over the test set: NDE, SAE, F1, and again with MAE. Equations (4.3), (4.4), and (4.9) describe them, with y_i and $\hat{y}_i$, $i = 1, \ldots, N$ being elements of the target and predicted signals respectively. MAE is explained in Section 2.2.1.

NDE, as the name says, indicates how well disaggregated the signal is. It reduces outlier influence at the target, and it can be used to choose a model that mimics well a target signature. SAE is a normalized discrepancy on energy estimations; it is especially useful if the total appliance consumption throughout a window is more important than the signal signature. Both NDE and SAE lie within the interval $[0, \infty)$, where the values closer to zero are better and values above 1.0 are considered extremely bad.

$$\text{NDE} = \frac{\sum_1^N \left(\hat{y}_i - y_i\right)^2}{\sum_1^N y_i^2} \tag{4.3}$$

$$\text{SAE} = \frac{\left|\sum_1^N \hat{y}_i - \sum_1^N y_i\right|}{\sum_1^N y_i} \tag{4.4}$$

F_1-Score is related to test accuracy in binary classification. Target and predicted signals are transformed into binary signals following Eq. (4.5), and they are then used in the logical functions shown in Eqs. (4.6), (4.7), and (4.8), where TP is true positives, FN is false negatives, and FP is false positives. Equation (4.5) is computed over the entire test set, instead of a window of 1024 data points. The aim is to verify if the model correctly classifies timestamps at which the appliances are ON or OFF. It is seen as a percentage of time during which the machine is correctly classified as turned ON or OFF. F1 is bounded in $[0, 1]$, where the closer to 1, the fewer the false events.

$$b\left(y\right)=\begin{cases}\text{ON}, & \text{if } y_i > \frac{\sum_1^N y_i}{N}\\ \text{OFF}, & \text{else}\end{cases} \tag{4.5}$$

$$\text{TP}=b(\hat{y})\wedge b(y) \tag{4.6}$$

$$\text{FN}=\neg b(\hat{y})\wedge b(y) \tag{4.7}$$

$$\text{FP}=b(\hat{y})\wedge \neg b(y) \tag{4.8}$$

$$\text{F}_1=\frac{2\cdot\text{TP}}{2\cdot\text{TP}+\text{FN}+\text{FP}} \tag{4.9}$$

Chapter 5

Results

The main results can be seen in Tab. 5.1. The FHMM and WaveNILM model comparison is based on F1, NDE, SAE, MAE values. It is possible to note how FHMM modeled EFI, EFII, DPI, and DPII appliances poorly. As seen in Figs. 5.1 to 5.16, appliances poorly modeled by FHMM switch states at almost every time step, thus inflicting low scores in F1 and high errors in NDE, SAE, and MAE. This poor modeling is probably due to the low power consumption they use, comparing to other appliances, as seen in Sec. 3.4. The pelletizers and milling machines are so important in the factory active power consumption that the electric noise that each of them generates is higher than the sum of the other appliances active power consumption. As FHMM only uses active power in its model inference – in its NILMTK implementation –, it might not have the necessary information to properly separate each appliance from the site meter active power consumption. Although the milling machines consume much active power, they are the appliances with the smallest number of available data points, which may have have impacted on how FHMM learns their signatures and states.

WaveNILM, on the other hand, uses current and voltage as inputs and has a high capability of learning time-series features. This difference results in less degradation for the smaller appliances such as DPI, EFI and EFII. The milling machines are also less affected by the smaller amount of data points available for training, in comparison to the number of points available for FHMM inference. F_1-Score shows that WaveNILM correctly infers, more often than the FHMM model, when each appliance is turned ON or OFF. Milling machines (seen in Figs. 5.15 and 5.16) active power oscillates during working hours, and the WaveNILM predicts this oscillation with 93.0 ± 0.9 and 93.8 ± 0.5 F_1-Score. Table 5.1 presents that WaveNILM models have small variation regarding training/validation fold.

Figures 5.9 to 5.16 help visualize how WaveNILM accurately predicts each appliance signal, including when it is turned on. We can see that noise is smoothed in all devices. Given the highly probabilistic nature of noise, electric noise is hardly modeled. In the point of view of the factory, noise can have some impact on its electric bill, especially for machines such as pelletizers, as electric noise can create fines due to high power demand – in case of switching noise – or request a tighter maintenance schedule. However, it does not create a serious problem with total factory energy consumption, and this can be viewed by SAE results. Smaller SAE is related to smoother noise.

DPII has the worst results in Tab. 5.1, and Fig. 5.12 shows that DPII model falsely predicts double-pole contactors signals in between time steps 2×10^4 and 6×10^4, but if we compare it with Fig. 5.11, then we can see that DPI is working at the same moment when DPII predicts it. Thus models do learn how to predict types of appliances and not individual appliances, even though the model DPII was never trained with DPI targets.

FHMM also yields its worst results while predicting DPII appliance. Nevertheless, unlike in the WaveNILM case, Figs. 5.3 and 5.4 show us both predictions are visually equally wrong, with many oscillations, and DPII gets worse scores and errors because it is OFF during more time steps than DPI, while both predicted signals are on 50% of the window frame. Those scores do not show up on other appliances, because only DPII seems to work differently during this test window.

If we analyze SAE, and thus the total energy consumption during this time frame, WaveNILM has less than 9% error on most appliances, while FHMM has at least 20% error for each non-pelletizer device. WaveNILM best-modeled appliances according to SAE are EFI and EFII with $0.9\% \pm 0.6\%$ and $1.1\% \pm 0.5\%$. If we compare these numbers with Figs. 5.13 and 5.14, we conclude that models predict the switching-state peaks well, and that low noise from the devices have some correlation with the low SAEs. In comparison, the WaveNILM pelletizer models do not predict any switching peak occurrence during this time frame, and get $2\% \pm 0.3\%$ and $3.7\% \pm 0.4\%$ errors. This trend is not exposed by NDE, and pelletizers models achieve better results than the exhaust fans models.

MAE cannot be compared explicitly for different appliances, but we can compare WaveNILM with FHMM models. Table 5.1 shows us that, for each appliance, WaveNILM has lower MAE than FHMM. If we compare this MAE column with Tab. 3.2, then we can see that WaveNILM models result in MAE compatible with the standard deviation of each appliance except for DPII.

Table 5.1: Comparative FHMM and WaveNILM results regarding F1, NDE, SAE, and MAE.

Appliances	F1 (×100)			NDE			SAE			MAE		
	FHMM	WaveNILM		FHMM	WaveNILM		FHMM	WaveNILM		FHMM	WaveNILM	
Pelletizer I	97.38	98.27	±0.05	0.040	0.024	±0.001	0.042	0.020	±0.003	4372	3015	±63
Pelletizer II	96.39	97.42	±0.06	0.063	0.0369	±0.0009	0.091	0.037	±0.004	6089	4034	±47
Double-pole Contactor I	74.09	97.17	±0.04	0.424	0.053	±0.004	0.326	0.07	±0.02	463	79	±8
Double-pole Contactor II	57.85	75.72	±0.05	0.626	0.68	±0.04	0.521	0.66	±0.05	576	286	±14
Exhaust Fan I	79.75	98.30	±0.03	0.378	0.0483	±0.0006	0.182	0.009	±0.006	1070	135	±1
Exhaust Fan II	78.38	98.1	±0.3	0.393	0.049	±0.007	0.200	0.011	±0.005	2008	248	±27
Milling Machine I	76.71	93.0	±0.9	0.335	0.11	±0.02	0.433	0.08	±0.05	14608	4317	±428
Milling Machine II	75.83	93.8	±0.5	0.286	0.078	±0.004	0.337	0.06	±0.02	11746	3154	±159

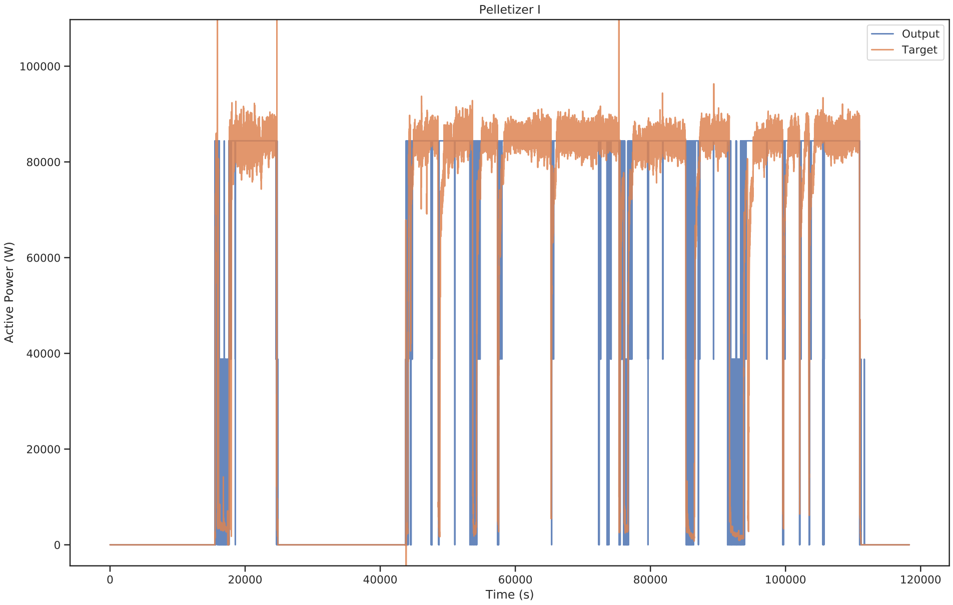


Figure 5.1: Pelletizer I FHMM model target and predicted signals. Results: F1: 97.38; NDE: 0.040; SAE: 0.042; MAE: 4372

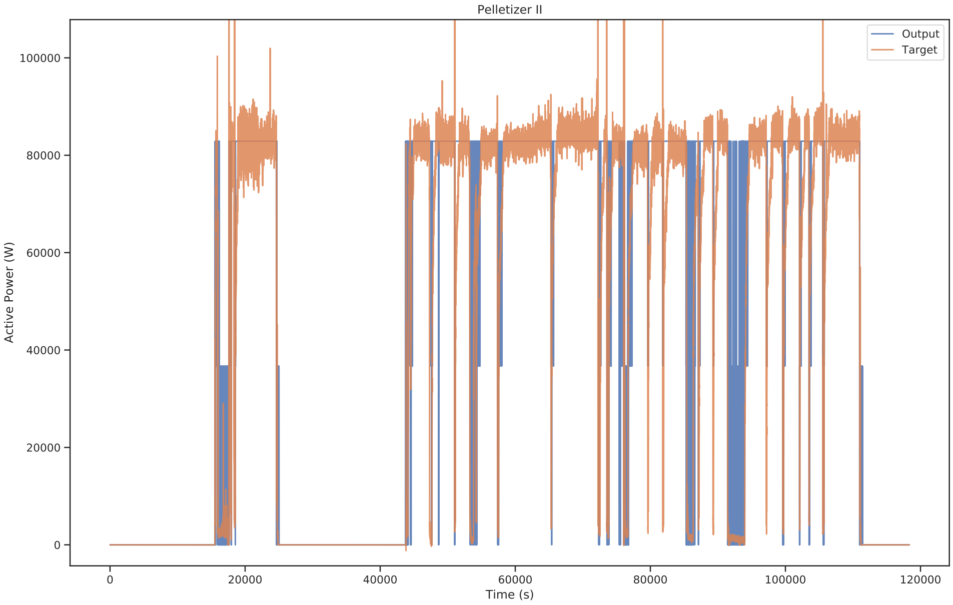


Figure 5.2: Pelletizer II FHMM model target and predicted signals. Results: F1: 96.39; NDE: 0.063; SAE: 0.091; MAE: 6089

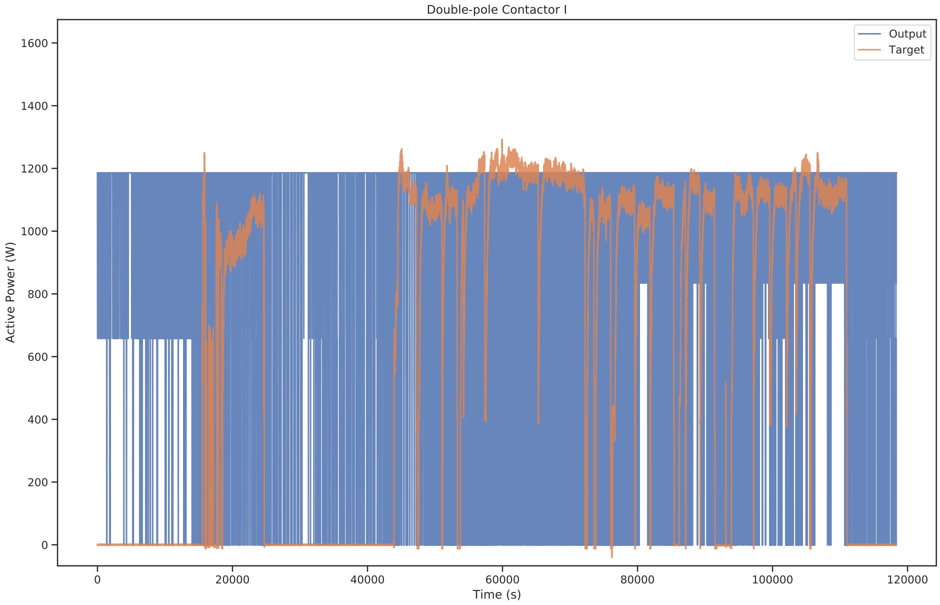


Figure 5.3: Double Pole Contactor I FHMM model target and predicted signals. Results: F1: 74.09; NDE: 0.424; SAE: 0.326; MAE: 463

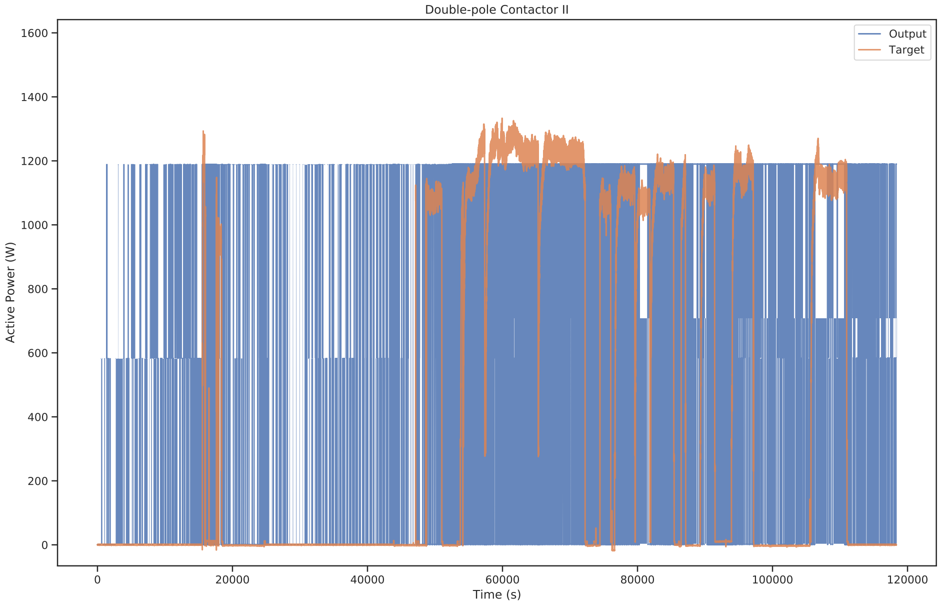


Figure 5.4: Double Pole Contactor II FHMM model target and predicted signals. Results: F1: 57.85; NDE: 0.626; SAE: 0.521; MAE: 576

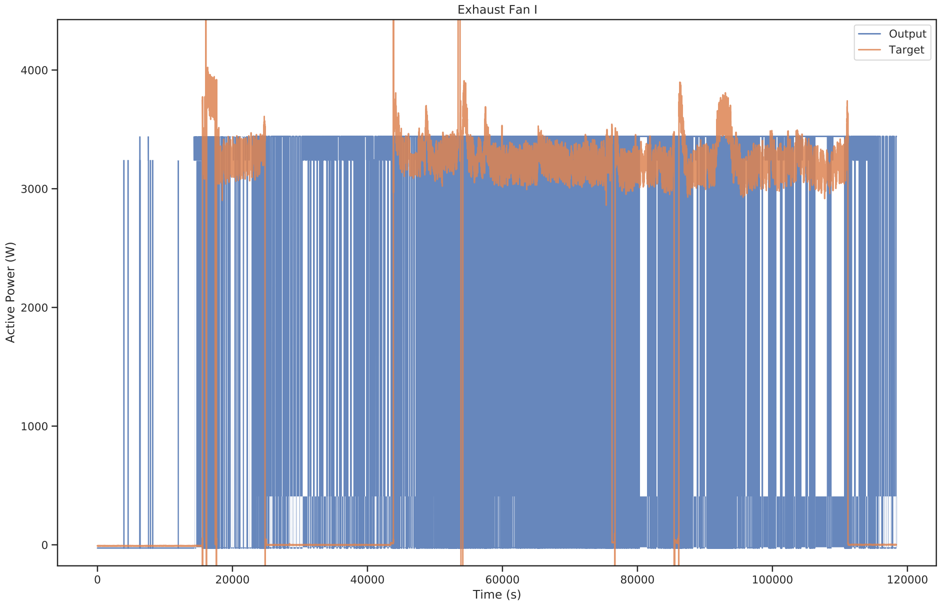


Figure 5.5: Exhaust Fan I FHMM model target and predicted signals. Results: F1: 79.75; NDE: 0.378; SAE: 0.182; MAE: 1070

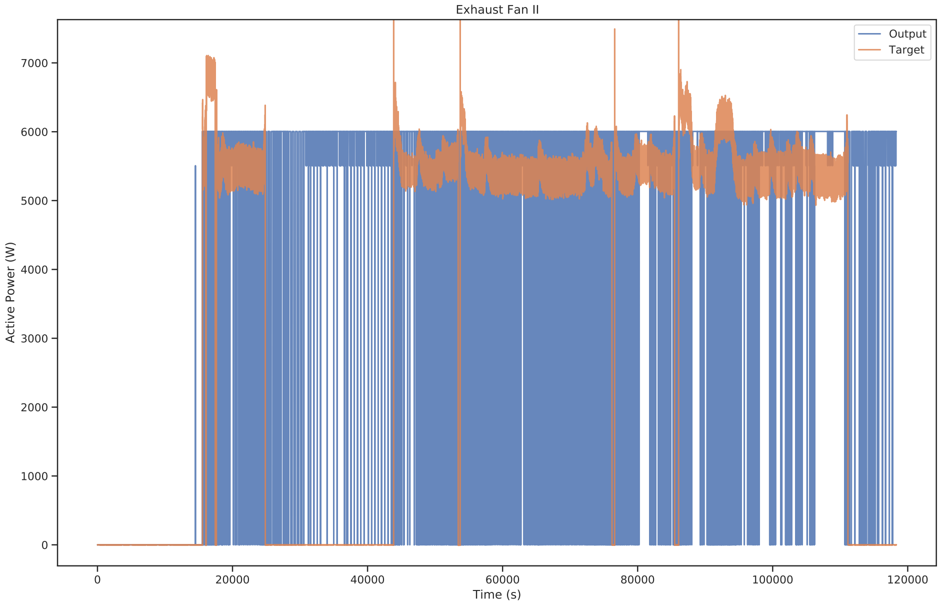


Figure 5.6: Exhaust Fan II FHMM model target and predicted signals. Results: F1: 78.38; NDE: 0.393; SAE: 0.200; MAE: 2008

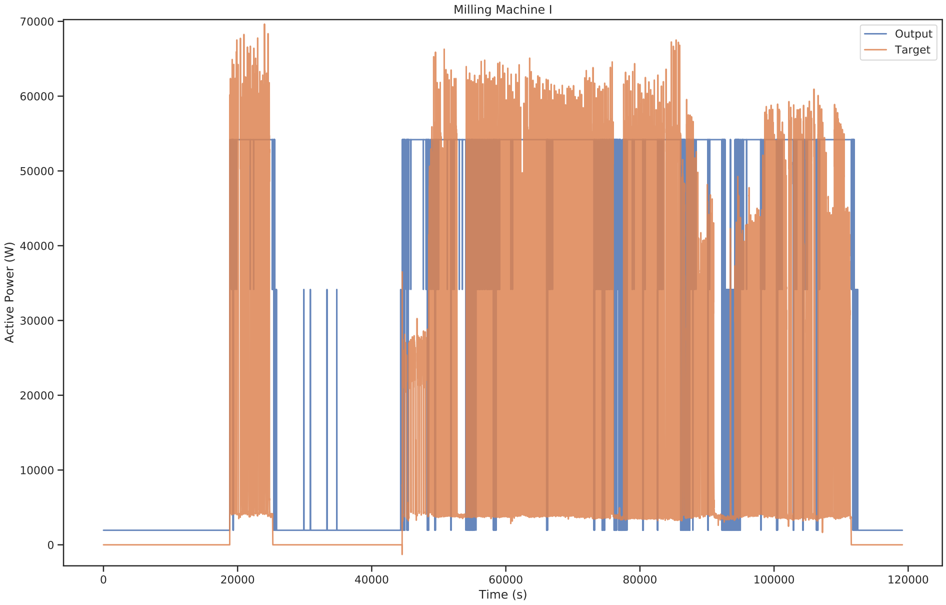


Figure 5.7: Milling Machine I FHMM model target and predicted signals. Results: F1: 76.71; NDE: 0.335; SAE: 0.433; MAE: 14608

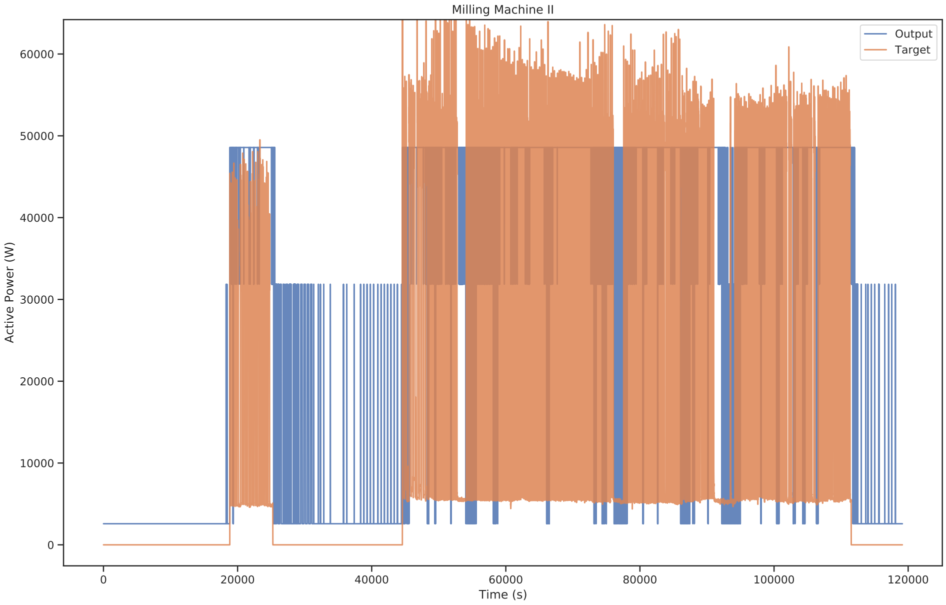


Figure 5.8: Milling Machine II FHMM model target and predicted signals. Results: F1: 75.83; NDE: 0.286; SAE: 0.337; MAE: 11746

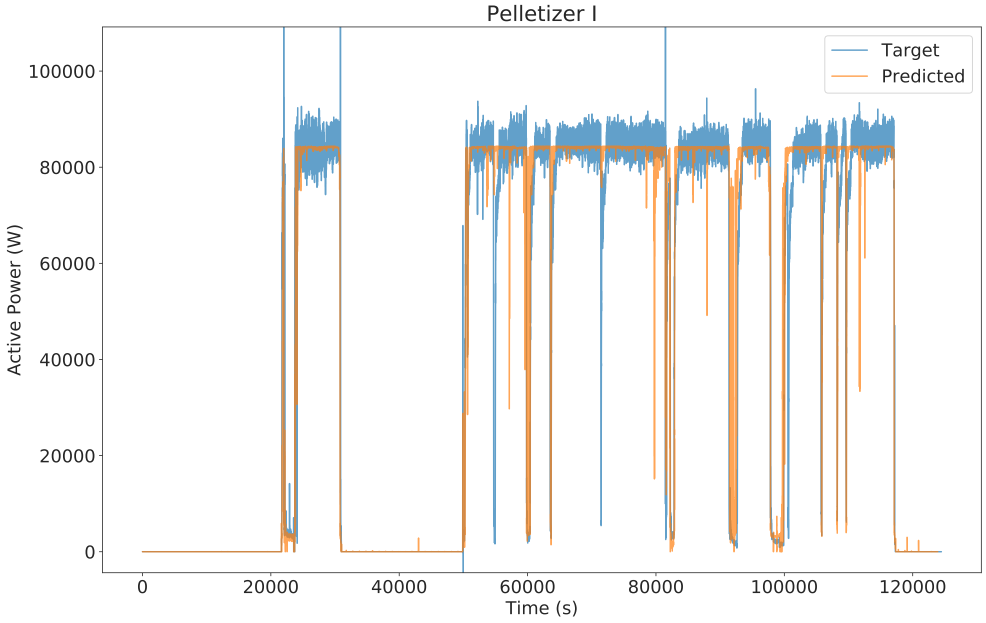


Figure 5.9: Pelletizer I WaveNILM model target and predicted signals. Results: F1: 98.27 ± .005; NDE: 0.024 ± 0.001; SAE: 0.020 ± 0.003; MAE: 3015 ± 63

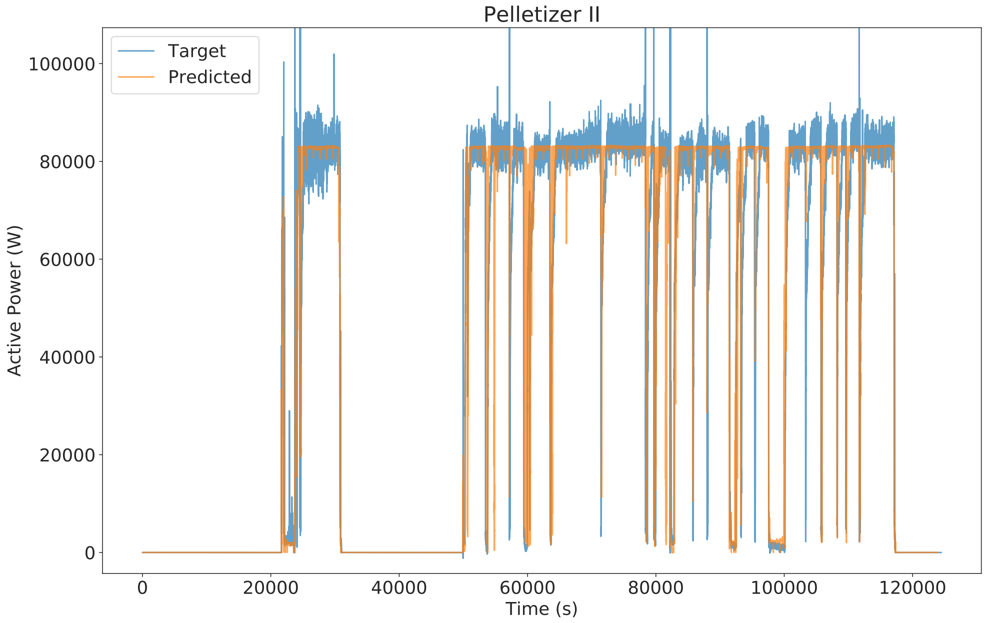


Figure 5.10: Pelletizer II WaveNILM model target and predicted signals. Results: F1: 97.42 ± 0.006; NDE: 0.0369 ± 0.0009; SAE: 0.037 ± 0.004; MAE: 4034 ± 47

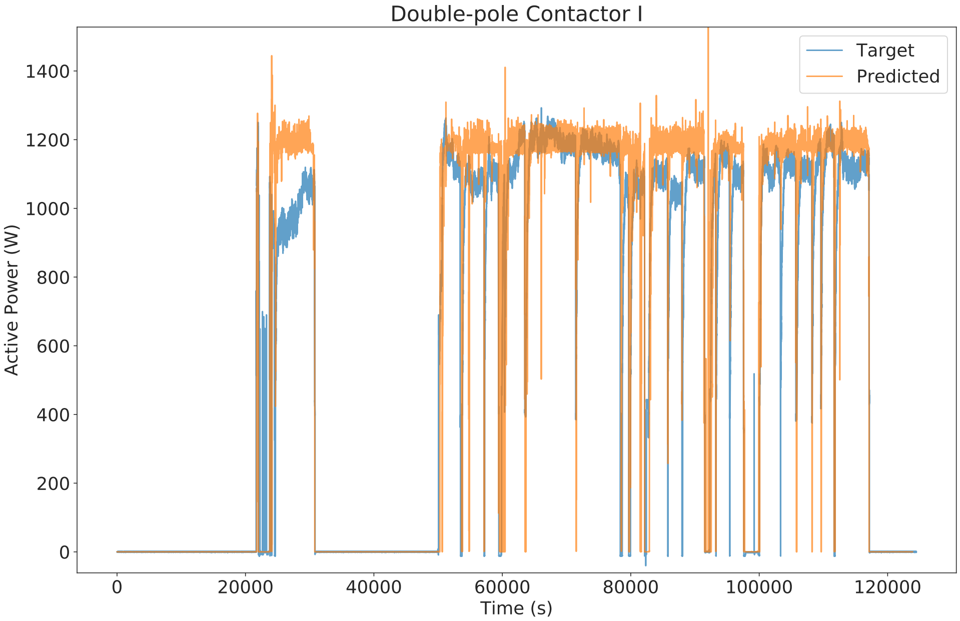


Figure 5.11: Double Pole Contactor I WaveNILM model target and predicted signals. Results: F1: 97.17 ± 0.04; NDE: 0.053 ± 0.004; SAE: 0.07 ± 0.02; MAE: 79 ± 8

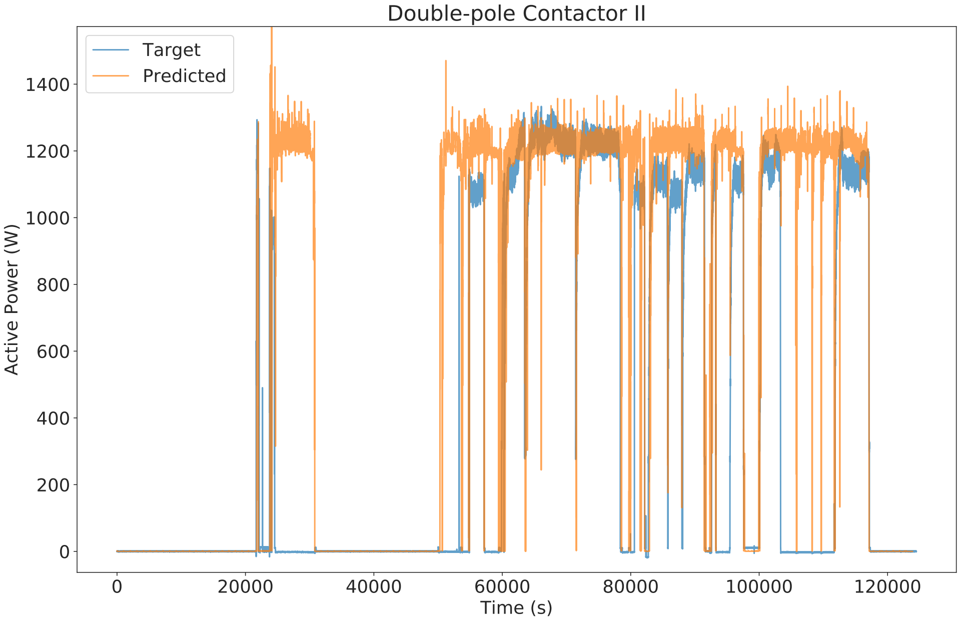


Figure 5.12: Double Pole Contactor II WaveNILM model target and predicted signals. Results: F1: 75.72 ± 0.05; NDE: 0.68 ± 0.04; SAE: 0.66 ± 0.05; MAE: 286 ± 14

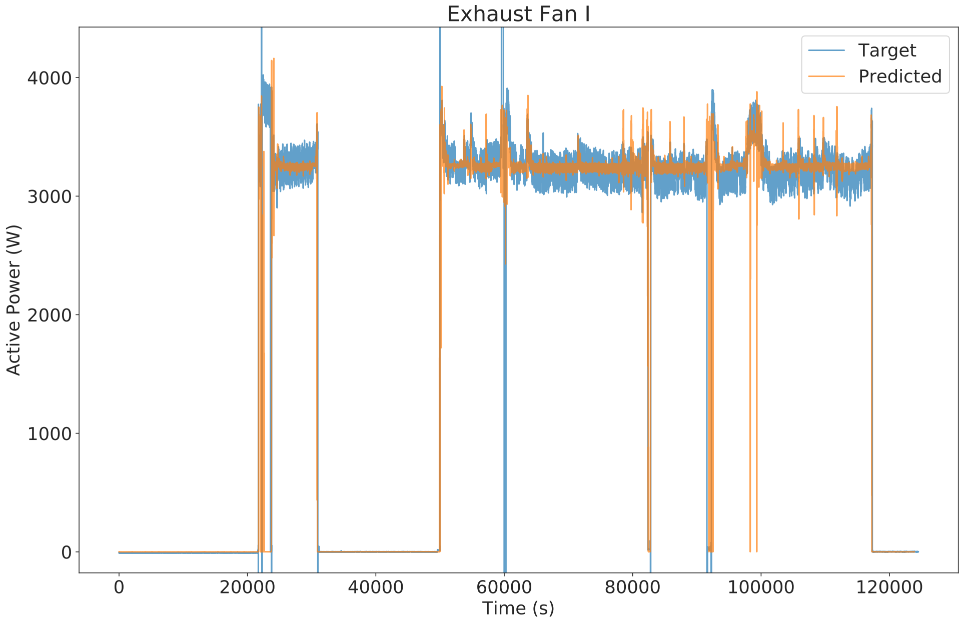


Figure 5.13: Exhaust Fan I WaveNILM model target and predicted signals. Results: F1: 98.30 ± 0.03; NDE: 0.0483 ± 0.0006; SAE: 0.009 ± 0.006; MAE: 135 ± 1

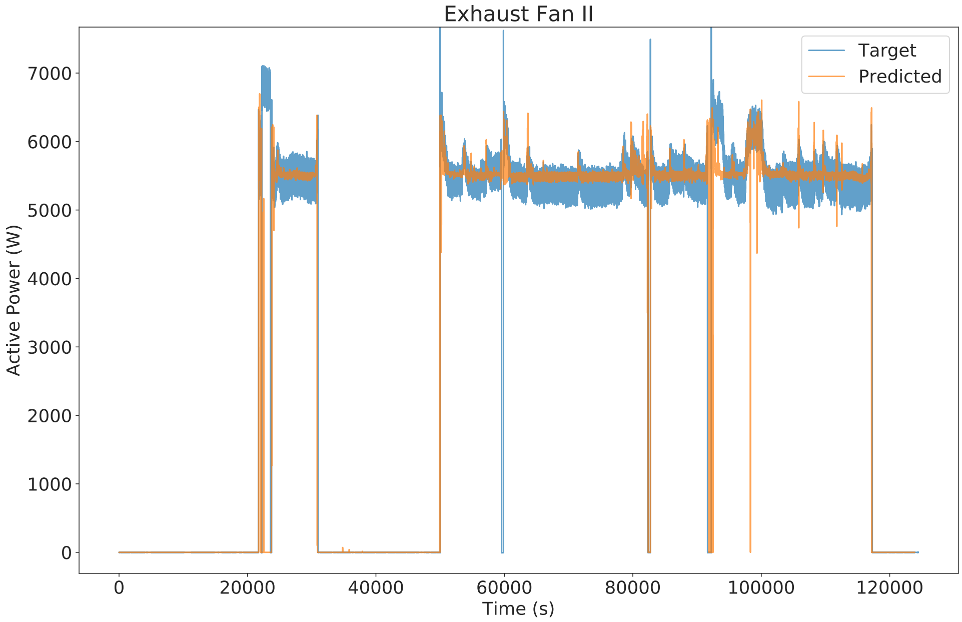


Figure 5.14: Exhaust Fan II WaveNILM model target and predicted signals. Results: F1: 98.1 ± 0.3; NDE: 0.049 ± 0.007; SAE: 0.011 ± 0.005; MAE: 248 ± 27

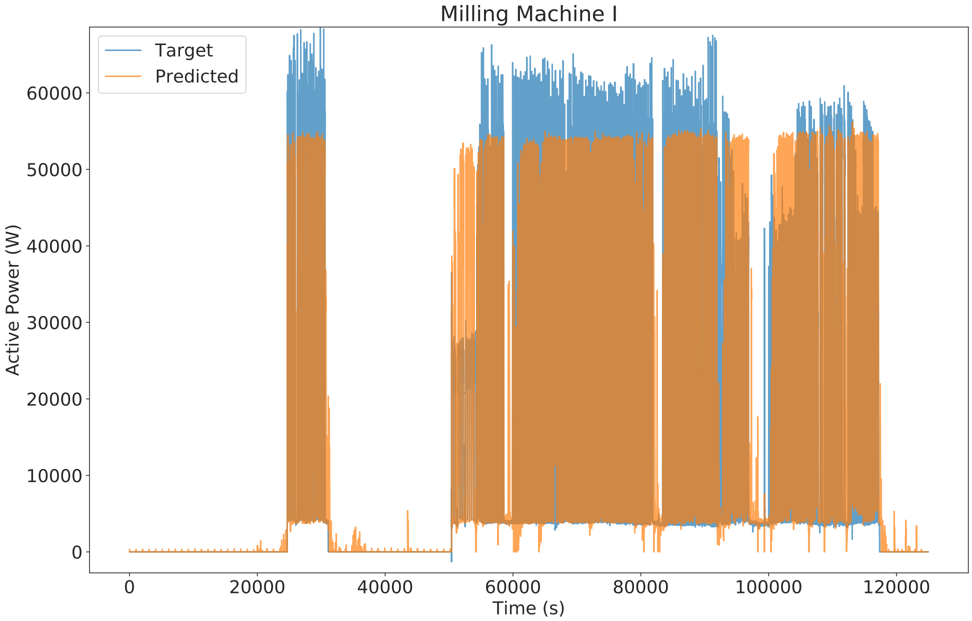


Figure 5.15: Milling Machine I WaveNILM model target and predicted signals. Results: F1: 93.0 ± 0.9; NDE: 0.11 ± 0.02; SAE: 0.08 ± 0.05; MAE: 4317 ± 428

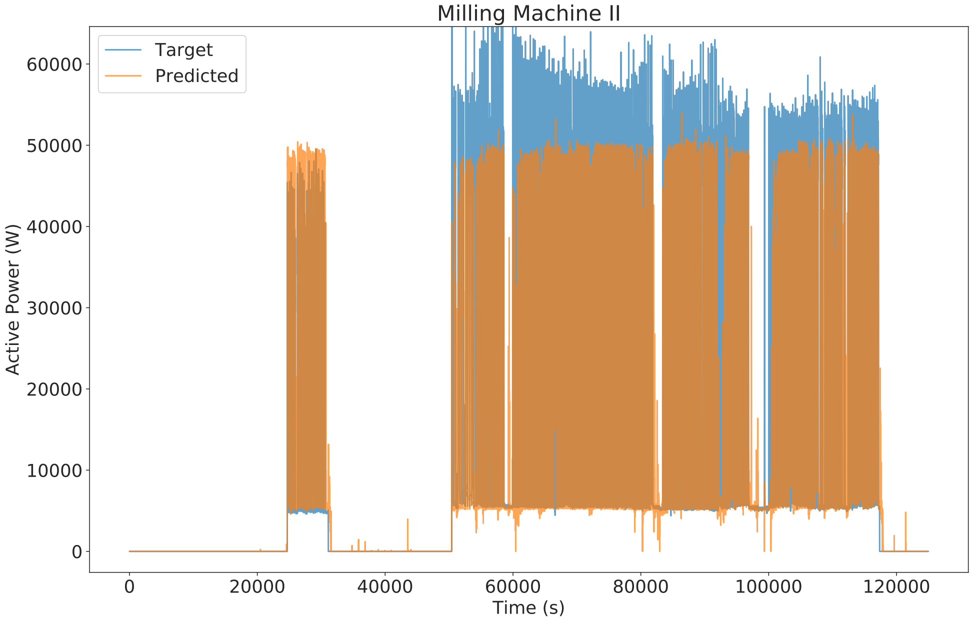


Figure 5.16: Milling Machine II WaveNILM model target and predicted signals. Results: F1: 93.8 ± 0.5; NDE: 0.078 ± 0.004; SAE: 0.06 ± 0.02; MAE: 3154 ± 159

Chapter 6

Conclusion

This work presents a new data set and a new attention-based model inspired by WaveNet called WaveNILM. It then compares WaveNILM to a standard FHMM model implemented by [8]. Both models are trained and tested using the data set presented here. The WaveNILM model shows that it is possible to replicate industry machinery electrical signature with only site-meter voltage and current, without using an event-based NILM approach or modeling each appliance as state machines. This result is possible because of the high capability of TCNs to model time-series signals. The factory also accepted to make the IMDELD data set publicly available for free at [1], which can be a great help for future investigations on the field. In this chapter, we present some conclusions and future works.

This work shows that the use of TCNs in NILM for industrial machines is an improvement in contrast with FHMM models. Also, the first public heavy-machinery data set for NILM is presented and detailed. Unfortunately, it was impossible to collect data from more factories, so all models are tested on an intra-building level. This means that these models are not verified to work on other industries, and the models may have represented the data with local overfit. When used in this specific scenario, this can be beneficial; if models stop disaggregating their appliances, then it is a sign that something has happened on the factory: a machine may have been turned permanently off, or it is malfunctioning.

WaveNILM achieve better results in all measurements when compared with default FHMM implemented by NILMTK. Based on DPI and DPII analysis, WaveNILM seems to learn and predict types of appliances instead of unique devices. This modeling implies that if two equal machines work at different time schedules, then one of them or both might not be adequately disaggregated. FHMM has results comparable to WaveNILM for machines with high load and a significant amount of data, but the low energy consumption devices and devices with a low amount of data are often modeled poorly. WaveNILM works well for devices with low power consumption and also with a smaller amount of data compared to the other appliances, though the latter may have been influenced by pre-training. With pre-training, WaveNILM models are able to train on a much bigger set of voltage/current data, and although the models are trained from different machines, trained models that are fined-tuned for each appliance achieves faster convergence and requires smaller data sets for training.

HOLMEGAARD and KJAERGAARD [22] assume that we cannot properly disaggregate loads with only one site-meter, and use sub-metering to achieve better results. In this work we show that it is possible to disagreggate even small machines (like EFI, EFII, DPI, and DPII) from the site meter. WaveNILM is capable to predict low consumption machines from the site-meter with same error range as the biggest and most important motors (PI and PII). Our approach does not depend on events detectors or stable steady-states, WaveNILM models aim at simulating a specific machine electrical signature from a site-meter, and they assume that all machines are CVDs. In contrast, FHMM assumes that all machines are FSMs. This can be troublesome for machines like milling machines, which do not have clearly distinguishable states, and are constantly increasing or decreasing power as they work.

6.1 Future work

We would like to encourage more industry data sets to be published, as it would help validate and test more models across different factories, industry sectors and machines. It would be interesting to find pelletizers, exhaust fans, double pole contactors, and milling machines with similar power consumption in other factories, in order to test if WaveNILM results keep consistent. It was not possible to measure machines without the interference of other appliances. It would be interesting to measure all those machines again in a controlled environment, where only they are turned on for a period of time. TCNs, and Neural Networks in general, are in a constant development state, and they should be tested in NILM applications. WaveNILM itself should also be tested with other topology parameters, for example, more stacks, or different receptive field sizes. As noted within residential householders by [59], after 52 weeks of measurements, user interest in appliance-level monitoring decreased by a factor of 90% average. A long-term study focusing on factory usage of appliance-level monitoring could be carried out, in order to clarify the usefulness of such monitoring in this sector.

Bibliography

[1] BANDEIRA DE MELLO MARTINS, P., BARBOSA NASCIMENTO, V., DE FREITAS, A. R., et al. “Industrial Machines Dataset for Electrical Load Disaggregation”. 2018. Available: <`http://dx.doi.org/10.21227/cg5v-dk02`>.

[2] HART, G. W. “Nonintrusive appliance load monitoring”, *Proceedings of the IEEE*, v. 80, n. 12, pp. 1870–1891, 1992.

[3] JONES, E., OLIPHANT, T., PETERSON, P., et al. “SciPy: Open source scientific tools for Python”. 2001–. Available: <`http://www.scipy.org/`>. [Online; accessed 09/17/2020].

[4] VAN DER WALT, S., COLBERT, S. C., VAROQUAUX, G. “The NumPy Array: A Structure for Efficient Numerical Computation”, *Computing in Science Engineering*, v. 13, n. 2, pp. 22–30, March 2011. ISSN: 1521-9615. doi: 10.1109/MCSE.2011.37.

[5] MCKINNEY, W. “Data Structures for Statistical Computing in Python”. In: van der Walt, S., Millman, J. (Eds.), *Proceedings of the 9th Python in Science Conference*, pp. 51 – 56, 2010.

[6] ABADI, M., AGARWAL, A., BARHAM, P., et al. “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems”. 2015. Available: <`http://tensorflow.org/`>. Software available from tensorflow.org.

[7] CHOLLET, F. “keras”. `https://github.com/fchollet/keras`, 2015.

[8] BATRA, N., KELLY, J., PARSON, O., et al. “NILMTK: an open source toolkit for non-intrusive load monitoring”. In: *Proceedings of the 5th international conference on Future energy systems*, pp. 265–276. ACM, 2014.

[9] IEA, I. E. A. *World Energy Balances 2018*. 2018. doi: https://doi.org/https://doi.org/10.1787/world_energy_bal-2018-en. Available: <`https://www.oecd-ilibrary.org/content/publication/world_energy_bal-2018-en`>.

[10] FREDERIKS, E. R., STENNER, K., HOBMAN, E. V. "Household energy use: Applying behavioural economics to understand consumer decision-making and behaviour", *Renewable and Sustainable Energy Reviews*, v. 41, pp. 1385 – 1394, 2015. ISSN: 1364-0321. doi: https://doi.org/10.1016/j.rser.2014.09.026. Available: <`http://www.sciencedirect.com/science/article/pii/S1364032114007990`>.

[11] ARMEL, K. C., GUPTA, A., SHRIMALI, G., et al. "Is disaggregation the holy grail of energy efficiency? The case of electricity", *Energy Policy*, v. 52, pp. 213–234, 2013.

[12] DARBY, S., OTHERS. "The effectiveness of feedback on energy consumption", *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, v. 486, n. 2006, pp. 26, 2006.

[13] ZEIFMAN, M., ROTH, K. "Nonintrusive appliance load monitoring: Review and outlook", *IEEE Transactions on Consumer Electronics*, v. 57, n. 1, pp. 76–84, February 2011. ISSN: 0098-3063. doi: 10.1109/TCE.2011.5735484.

[14] KELLY, J., KNOTTENBELT, W. "Neural NILM: Deep Neural Networks Applied to Energy Disaggregation". In: *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, pp. 55–64, New York, NY, USA, 2015. ACM. ISBN: 978-1-4503-3981-0. doi: 10.1145/2821650.2821672. Available: <`http://doi.acm.org/10.1145/2821650.2821672`>.

[15] PARSON, O., GHOSH, S., WEAL, M., et al. "Using hidden Markov models for iterative non-intrusive appliance monitoring". In: *Proceedings of the 25th Annual Conference on Neural Information Processing Systems workshop on Machine Learning for Sustainability*, Granada, Spain, December 2011. Available: <`https://eprints.soton.ac.uk/272990/`>. Event Dates: 17 December 2011.

[16] KOLTER, J. Z., JAAKKOLA, T. "Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation". In: Lawrence, N. D., Girolami, M. (Eds.), *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, v. 22, *Proceedings of Machine Learning Research*, pp. 1472–1482, La Palma, Canary Islands, 21–23 Apr 2012. PMLR. Available: <`http://proceedings.mlr.press/v22/zico12.html`>.

[17] HE, K., STANKOVIC, L., LIAO, J., et al. "Non-intrusive load disaggregation using graph signal processing", *IEEE Transactions on Smart Grid*, 2017.

[18] ZHANG, C., ZHONG, M., WANG, Z., et al. "Sequence-to-point learning with neural networks for nonintrusive load monitoring", *arXiv preprint arXiv:1612.09106*, 2016.

[19] BATRA, N., PARSON, O., BERGES, M., et al. "A comparison of non-intrusive load monitoring methods for commercial and residential buildings", *arXiv preprint arXiv:1408.6595*, 2014.

[20] DO NASCIMENTO, P. P. M. *Applications of deep learning techniques on NILM*. Master's thesis, PEE/COPPE - Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, 2016.

[21] MORGAN, E. S. *Applications of Deep Learning for Load Disaggregation in Residential Environments*. Bachelor's thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, 2017.

[22] HOLMEGAARD, E., KJAERGAARD, M. B. "NILM in an Industrial Setting: A Load Characterization and Algorithm Evaluation". In: *Smart Computing (SMARTCOMP), 2016 IEEE International Conference on*, pp. 1–8, St. Louis, Missouri, USA, May 2016. IEEE.

[23] MARTINS, P. B. M., GOMES, J. G. R. C., NASCIMENTO, V. B., et al. "Application of a Deep Learning Generative Model to Load Disaggregation for Industrial Machinery Power Consumption Monitoring". In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1–6, Oct 2018. doi: 10.1109/SmartGridComm.2018.8587415.

[24] *Balanço Energético Nacional*. Technical report, Empresa de Pesquisa Energética, Ministério de Minas e Energia, 2017.

[25] RUANO, A., HERNANDEZ, A., UREÑA, J., et al. "NILM Techniques for intelligent home energy management and ambient assisted living: A review", *Energies*, v. 12, n. 11, pp. 2203, 2019.

[26] E SILVA, P. B., DE SEIXAS, J. M., CALÔBA, L. P. "Identificação de Cargas Domésticas Utilizando uma Rede ART a Partir de Amostras de Potência Ativa e Reativa". In: Braga, A. d. P., Bastos Filho, C. J. A. (Eds.), *Anais do 11 Congresso Brasileiro de Inteligência Computacional*, pp. 1–5, Porto de Galinhas, PE, 2013. SBIC.

[27] YANG, C. C., SOH, C. S., YAP, V. V. “Comparative study of event detection methods for non-intrusive appliance load monitoring”, *Energy Procedia*, v. 61, pp. 1840–1843, 2014.

[28] ZHAO, B., HE, K., STANKOVIC, L., et al. “Improving event-based non-intrusive load monitoring using graph signal processing”, *IEEE Access*, v. 6, pp. 53944–53959, 2018.

[29] LE, T.-T.-H., KIM, H., OTHERS. “Non-intrusive load monitoring based on novel transient signal in household appliances with low sampling rate”, *Energies*, v. 11, n. 12, pp. 3409, 2018.

[30] ANDERSON, K. D., BERGÉS, M. E., OCNEANU, A., et al. “Event detection for non intrusive load monitoring”. In: *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*, pp. 3312–3317. IEEE, 2012.

[31] GUPTA, S., REYNOLDS, M. S., PATEL, S. N. “ElectriSense: Single-point Sensing Using EMI for Electrical Event Detection and Classification in the Home”. In: *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, UbiComp ’10, pp. 139–148, New York, NY, USA, 2010. ACM. ISBN: 978-1-60558-843-8. doi: 10.1145/1864349.1864375. Available: <http://doi.acm.org/10.1145/1864349.1864375>.

[32] RODRIGUEZ-SILVA, A., MAKONIN, S. “Universal Non-Intrusive Load Monitoring (UNILM) Using Filter Pipelines, Probabilistic Knapsack, and Labelled Partition Maps”, *arXiv e-prints*, art. arXiv:1907.06299, Jul 2019.

[33] LESHNO, M., LIN, V. Y., PINKUS, A., et al. “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function”, *Neural networks*, v. 6, n. 6, pp. 861–867, 1993.

[34] CYBENKO, G. “Approximation by superpositions of a sigmoidal function”, *Mathematics of control, signals and systems*, v. 2, n. 4, pp. 303–314, 1989.

[35] TELGARSKY, M. “Benefits of depth in neural networks”, *arXiv preprint arXiv:1612.09106*, 2016.

[36] WILLMOTT, C. J., MATSUURA, K. “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance”, *Climate research*, v. 30, n. 1, pp. 79–82, 2005.

[37] CHAI, T., DRAXLER, R. R. “Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature”, *Geoscientific model development*, v. 7, n. 3, pp. 1247–1250, 2014.

[38] WERBOS, P. J. “Backpropagation through time: what it does and how to do it”, *Proceedings of the IEEE*, v. 78, n. 10, pp. 1550–1560, 1990.

[39] BENGIO, Y., SIMARD, P., FRASCONI, P. “Learning long-term dependencies with gradient descent is difficult”, *IEEE transactions on neural networks*, v. 5, n. 2, pp. 157–166, 1994.

[40] HOCHREITER, S., SCHMIDHUBER, J. “Long short-term memory”, *Neural computation*, v. 9, n. 8, pp. 1735–1780, 1997.

[41] OLAH, C. “Understanding LSTM Networks”. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2015.

[42] CHO, K., VAN MERRIËNBOER, B., GULCEHRE, C., et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”, *arXiv preprint arXiv:1406.1078*, 2014.

[43] BAI, S., KOLTER, J. Z., KOLTUN, V. “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling”, *arXiv preprint arXiv:1803.01271*, 2018.

[44] ESTEVÃO FILHO, R. D. M. “A study on Deep Convolutional Neural Networks for computer vision applications”. 2017.

[45] YU, F., KOLTUN, V. “Multi-Scale Context Aggregation by Dilated Convolutions”. 2015.

[46] RABINER, L., JUANG, B. “An introduction to hidden Markov models”, *ieee assp magazine*, v. 3, n. 1, pp. 4–16, 1986.

[47] GHAHRAMANI, Z., JORDAN, M. I. “Factorial hidden Markov models”. In: *Advances in Neural Information Processing Systems*, pp. 472–478, 1996.

[48] KOLTER, J. Z., JOHNSON, M. J. “REDD: A public data set for energy disaggregation research”. In: *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*, v. 25, pp. 59–62, 2011.

[49] KELLY, J., KNOTTENBELT, W. “The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes”, *Scientific data*, v. 2, pp. 150007, 2015.

[50] MONACCHI, A., EGARTER, D., ELMENREICH, W., et al. "GREEND: An energy consumption dataset of households in Italy and Austria". In: *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp. 511–516. IEEE, 2014.

[51] FILIP, A. "Blued: A fully labeled public dataset for event-based nonintrusive load monitoring research". In: *2nd Workshop on Data Mining Applications in Sustainability (SustKDD)*, p. 2012, 2011.

[52] KAHL, M., HAQ, A. U., KRIECHBAUMER, T., et al. "Whited-a worldwide household and industry transient energy data set". In: *3rd International Workshop on Non-Intrusive Load Monitoring*, 2016.

[53] PICON, T., MEZIANE, M. N., RAVIER, P., et al. "COOLL: Controlled on/off loads library, a public dataset of high-sampled electrical signals for appliance identification", *arXiv preprint arXiv:1611.05803*, 2016.

[54] BECKEL, C., KLEIMINGER, W., CICCHETTI, R., et al. "The ECO Data Set and the Performance of Non-intrusive Load Monitoring Algorithms". In: *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pp. 80–89, New York, NY, USA, 2014. ACM. ISBN: 978-1-4503-3144-9. doi: 10.1145/2674061.2674064. Available: <http://doi.acm.org/10.1145/2674061.2674064>.

[55] INCORPORATION, P. S. "Dataport". 2015. Available: <https://dataport.pecanstreet.org/>.

[56] GREENANT. *GreenAnt Smart Meter - Beta (GASM-B) Manual do Usuário Rev.1.2 - 11/2018*. Technical report, 2018. Available: <https://files.greenant.com.br/manuals/GASM-B-Manual-do-usuario.pdf>. [Online; accessed 09/17/2020].

[57] VAN DEN OORD, A., DIELEMAN, S., ZEN, H., et al. "WaveNet: A Generative Model for Raw Audio". 2016.

[58] KELLY, J., KNOTTENBELT, W. "Metadata for Energy Disaggregation". In: *The 2nd IEEE International Workshop on Consumer Devices and Systems (CDS 2014)*, Västerås, Sweden, jul. 2014. doi: 10.1109/COMPSACW.2014.97.

[59] PEREIRA, L., QUINTAL, F., BARRETO, M., et al. "Understanding the Limitations of Eco-feedback: A One-Year Long-Term Study". In: Holzinger,

A., Pasi, G. (Eds.), *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, pp. 237–255, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN: 978-3-642-39146-0.

Appendix A

Histogram for all turned on appliances in data set

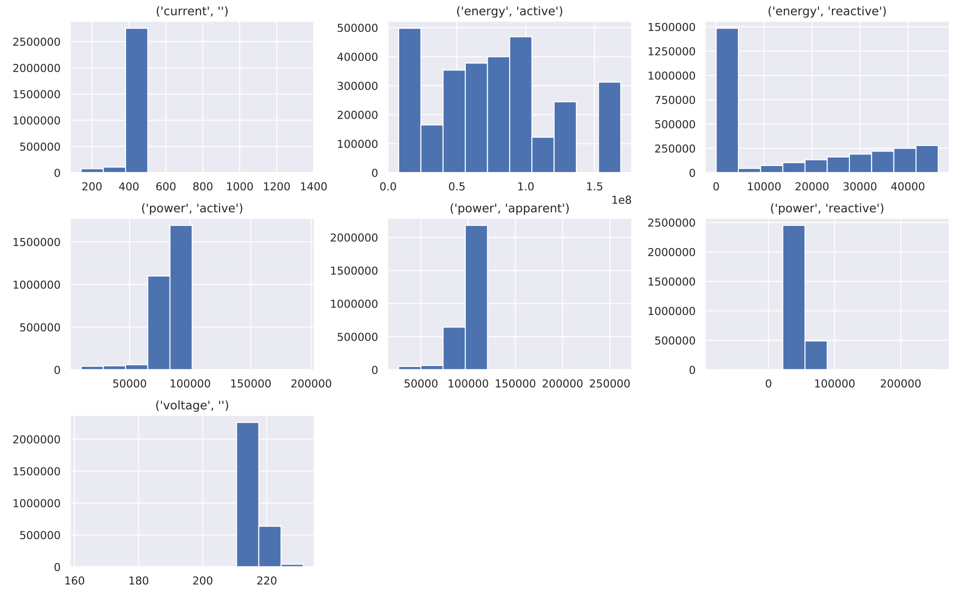


Figure A.1: Histogram of features from Pelletizer I.

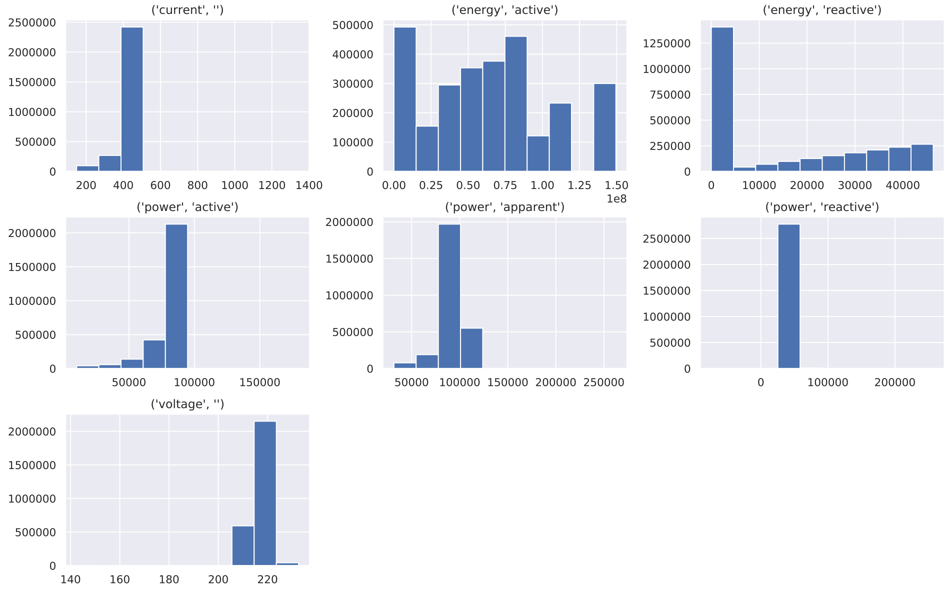


Figure A.2: Histogram of features from Pelletizer II.

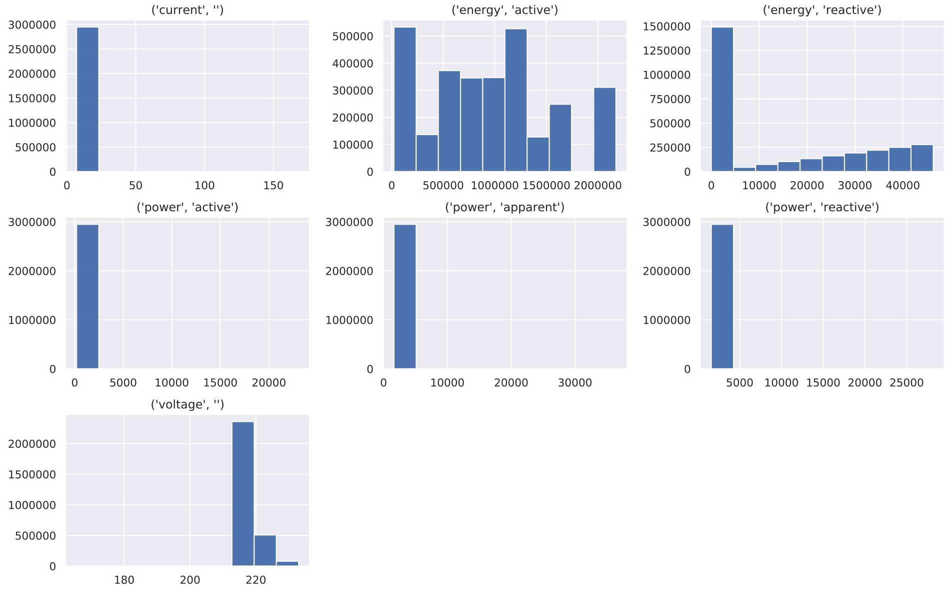


Figure A.3: Histogram of features from Double-pole contactor I.

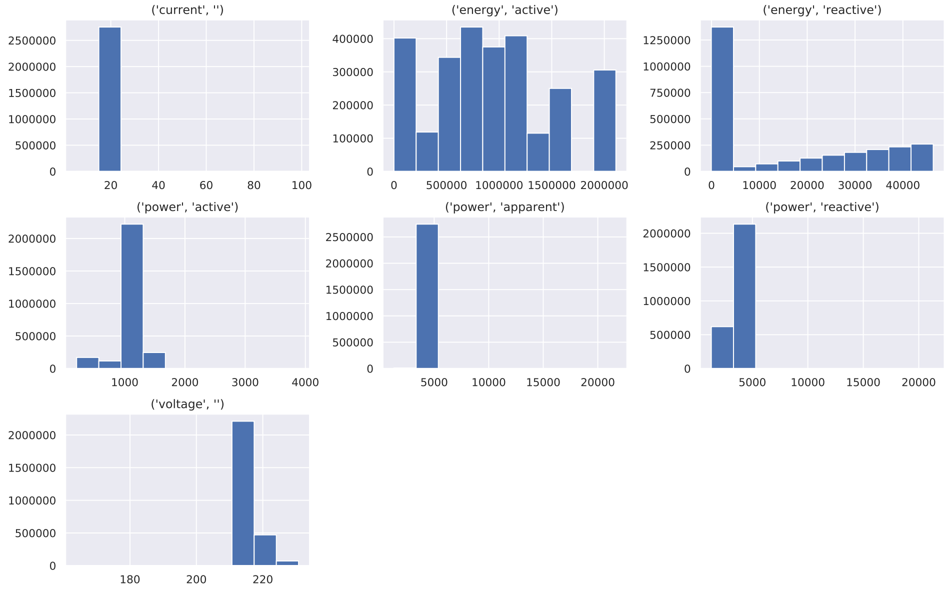


Figure A.4: Histogram of features from Double-pole contactor II.

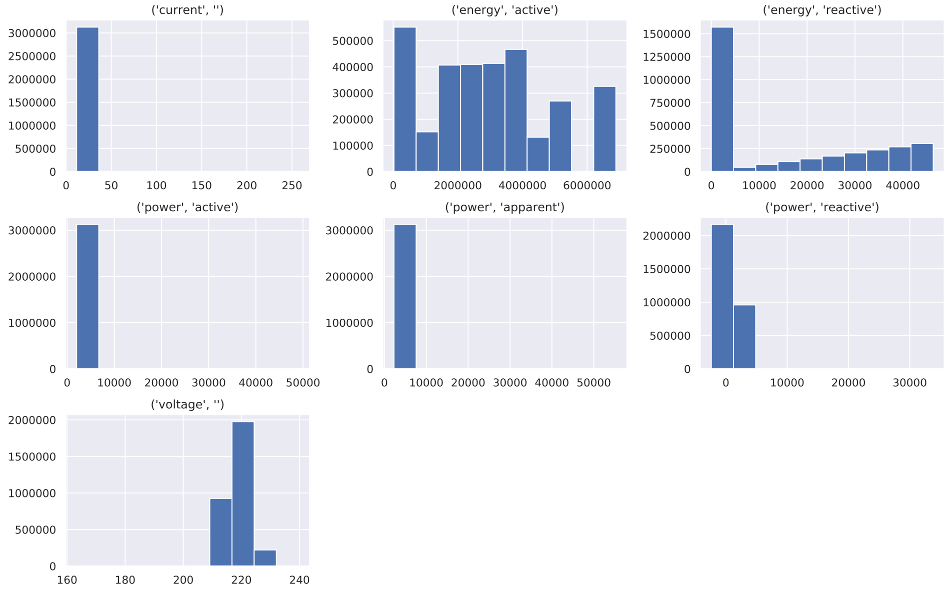


Figure A.5: Histogram of features from Exhaust Fan I.

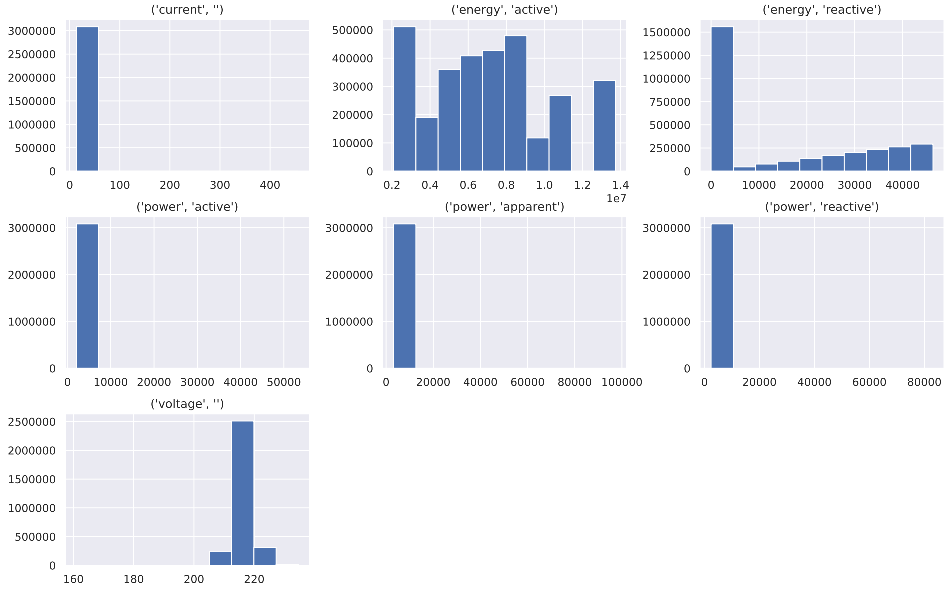


Figure A.6: Histogram of features from Exhaust Fan II.

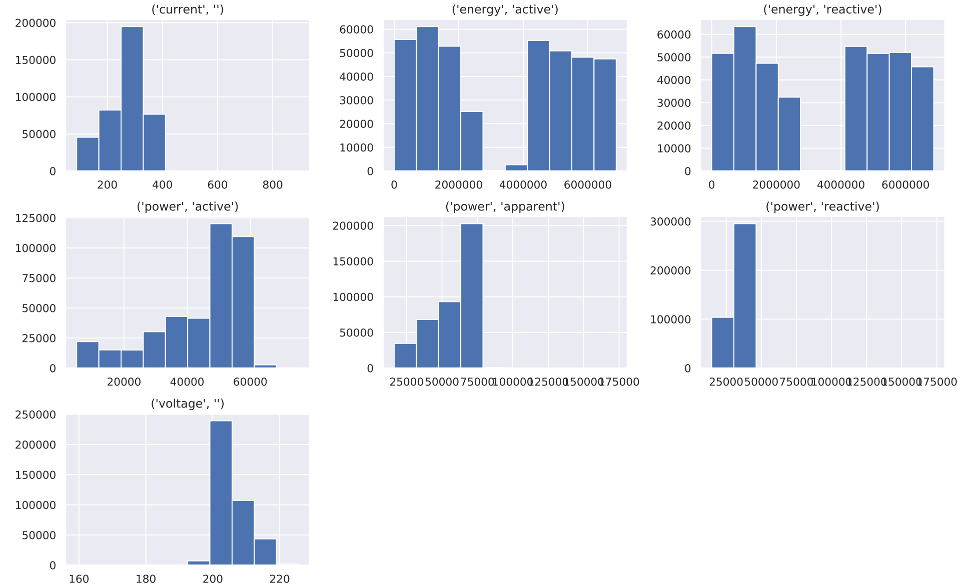


Figure A.7: Histogram of features from Milling Machine I.

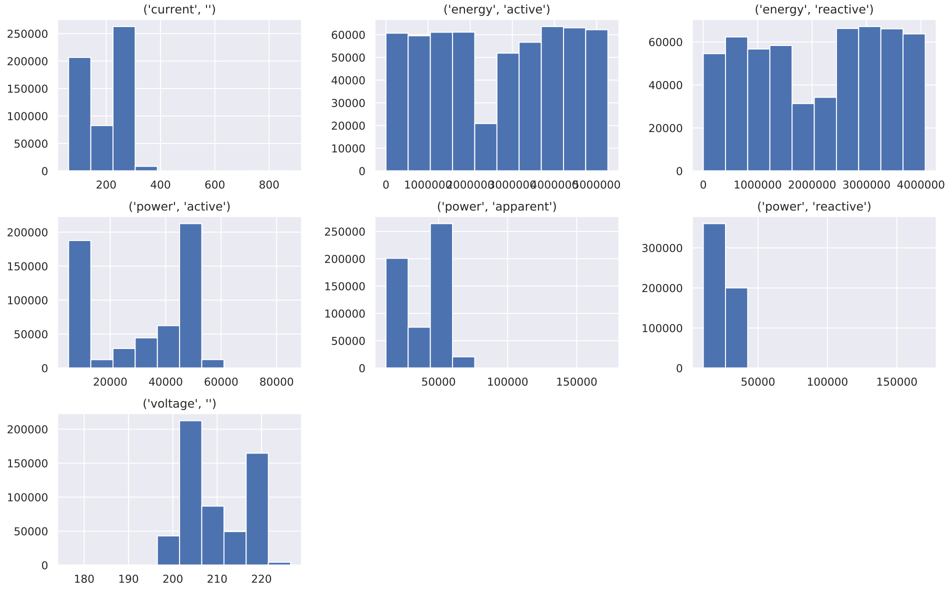


Figure A.8: Histogram of features from Milling Machine II.

Appendix B

Training and validation loss from WaveNILM training.

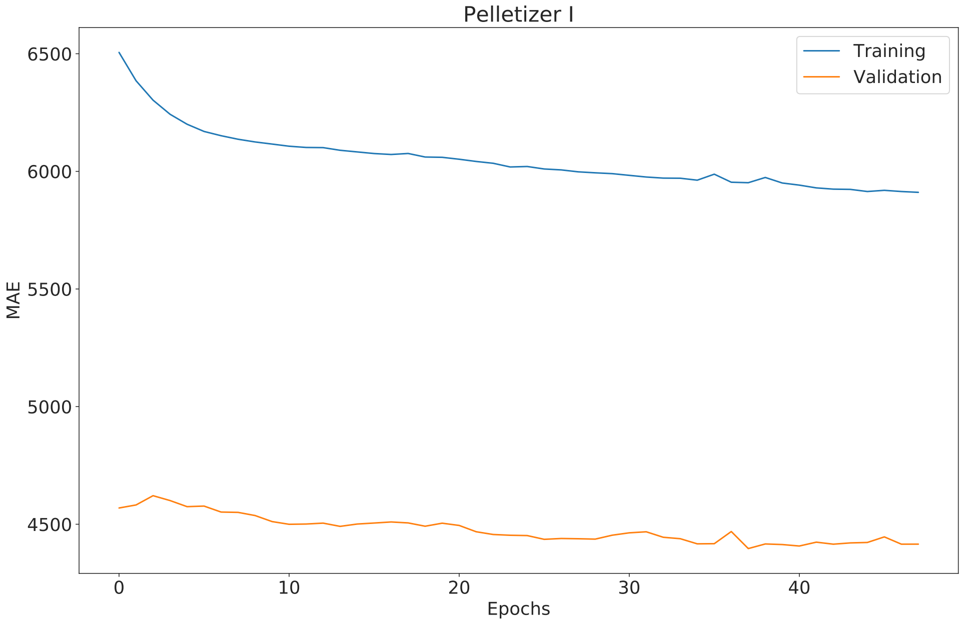


Figure B.1: Training loss and validation loss from Pelletizer I model training.

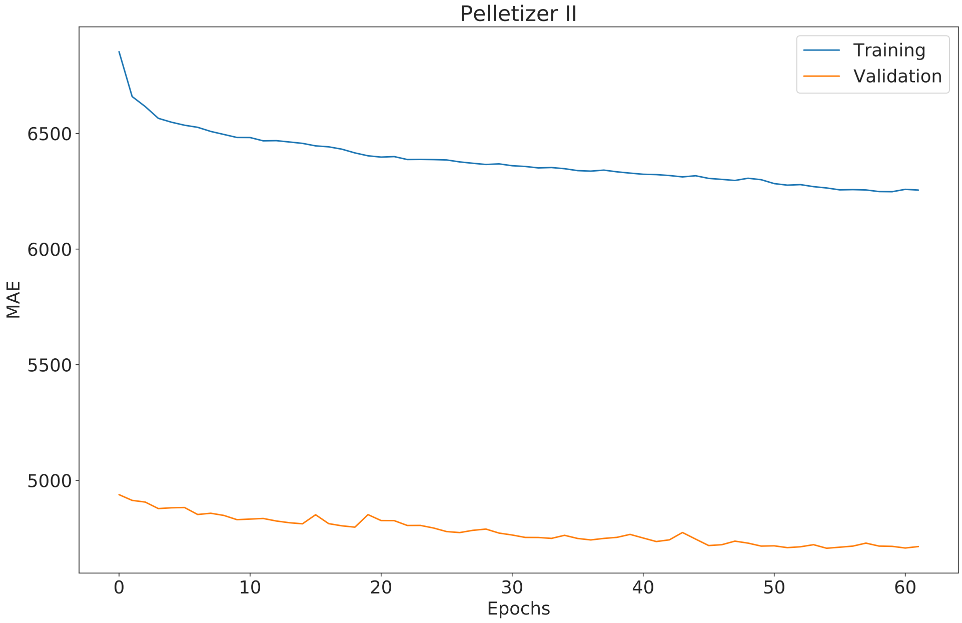


Figure B.2: Training loss and validation loss from Pelletizer II model training.

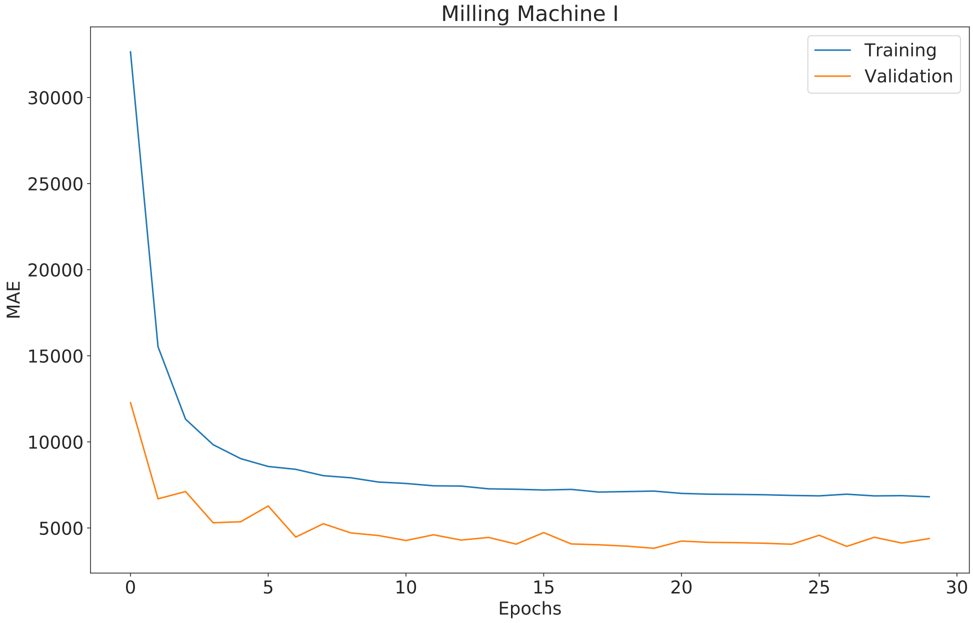


Figure B.3: Training loss and validation loss from Milling Machine I model training.

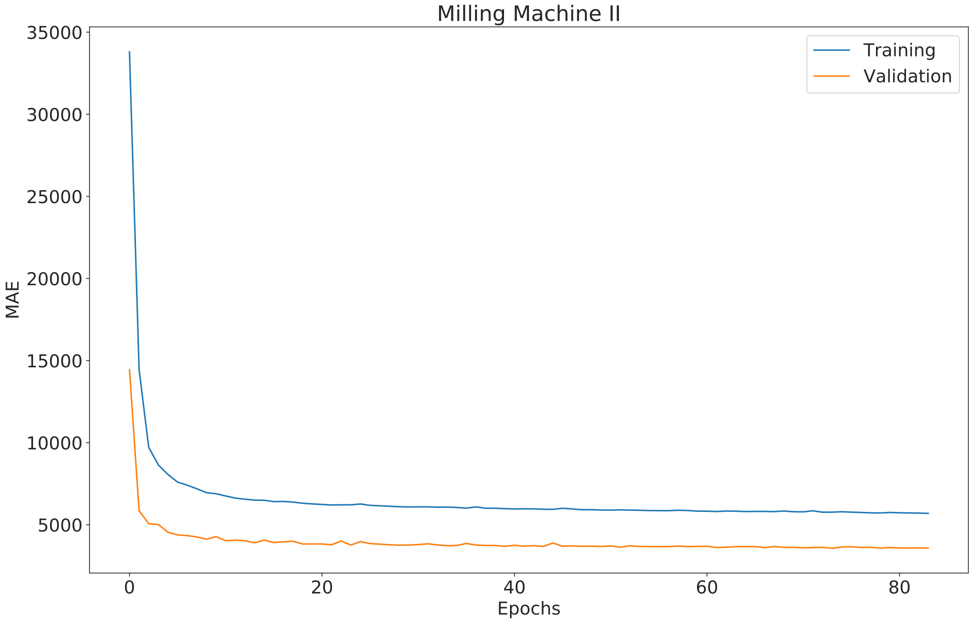


Figure B.4: Training loss and validation loss from Milling Machine II model training.

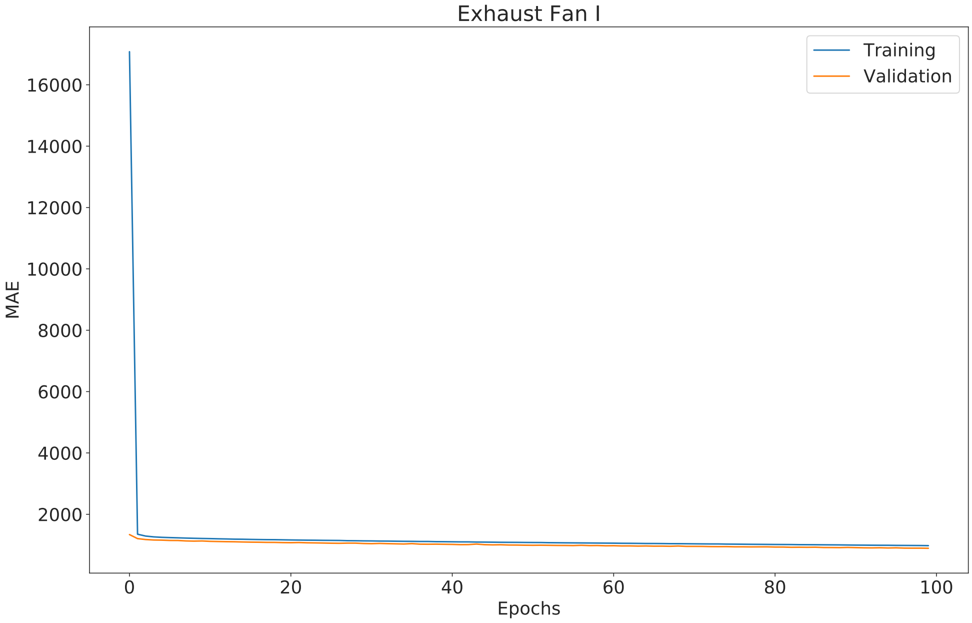


Figure B.5: Training loss and validation loss from Exhaust Fan I model training.

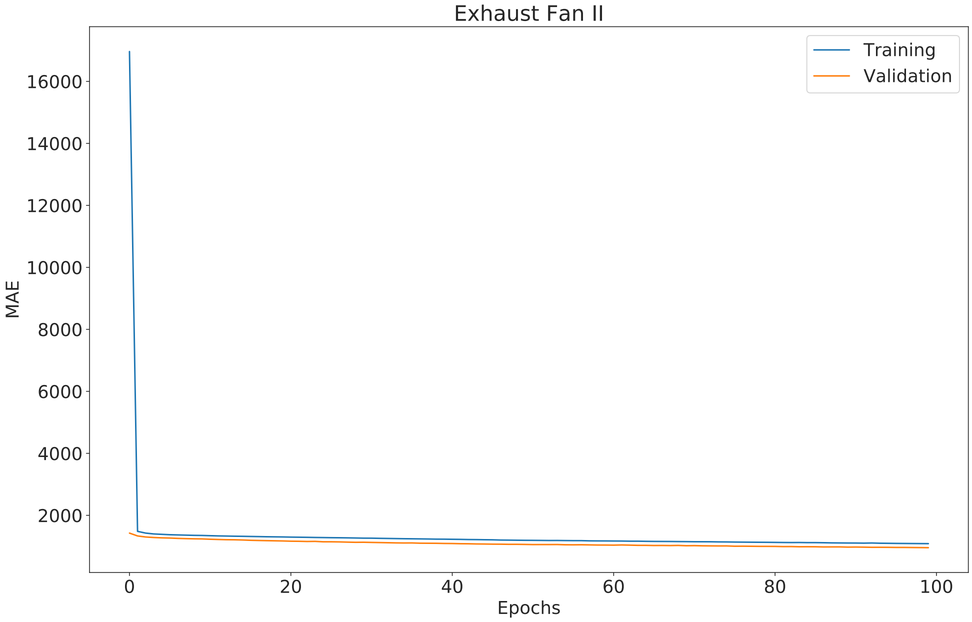


Figure B.6: Training loss and validation loss from Exhaust Fan II model training.

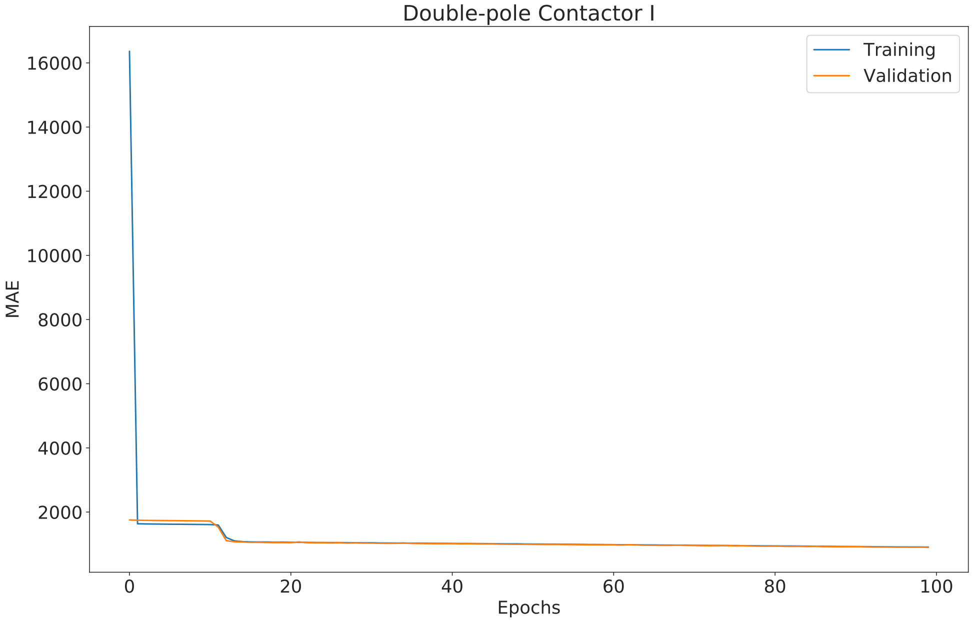


Figure B.7: Training loss and validation loss from Double Pole Contactor I model training.

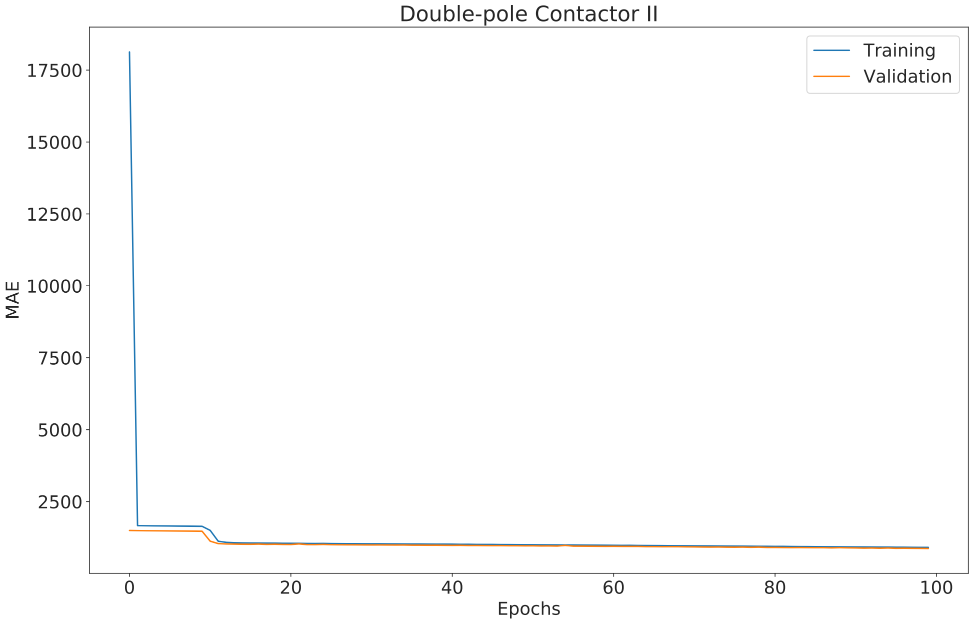


Figure B.8: Training loss and validation loss from Double Pole Contactor II model training.

Appendix C

Log file created during WaveNILM training

```
Window size: 1024
Number of epochs: 100
Number of Kfold splits: 7
Site_meter: MV/LV Transformer
Site_meter features: [('voltage', ''), ('current', '')]
Appliances: ['Pelletizer I', 'Pelletizer II', 'Double-pole Contactor I',
            'Double-pole Contactor II', 'Exhaust Fan I',
            'Exhaust Fan II', 'Milling Machine I', 'Milling Machine II']
Appliances features: [('power', 'active')]
Loss function: mae
Optimizer: Adam
Appliance: Pelletizer I on fold 0
   - train gap: 2017-12-11T20:51:15:2017-12-11T21:35:34
   - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:14
   - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 1
   - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:34
   - validation gap: 2017-12-11T20:51:15:2017-12-11T20:58:38
   - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 2
   - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:34
   - validation gap: 2017-12-11T20:58:40:2017-12-11T21:06:06
   - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 3
   - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:34
```

```
  - validation gap: 2017-12-11T21:06:07:2017-12-11T21:13:28
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:34
  - validation gap: 2017-12-11T21:13:29:2017-12-11T21:20:50
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:34
  - validation gap: 2017-12-11T21:20:51:2017-12-11T21:28:12
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer I on fold 6
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:28:12
  - validation gap: 2017-12-11T21:28:13:2017-12-11T21:35:34
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 0
  - train gap: 2017-12-11T20:51:20:2017-12-11T21:35:55
  - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:19
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 1
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:55
  - validation gap: 2017-12-11T20:51:20:2017-12-11T20:58:45
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 2
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:55
  - validation gap: 2017-12-11T20:58:46:2017-12-11T21:06:11
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 3
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:55
  - validation gap: 2017-12-11T21:06:12:2017-12-11T21:13:37
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:55
  - validation gap: 2017-12-11T21:13:38:2017-12-11T21:21:03
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:55
  - validation gap: 2017-12-11T21:21:04:2017-12-11T21:28:29
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Pelletizer II on fold 6
```

```
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:28:29
  - validation gap: 2017-12-11T21:28:30:2017-12-11T21:35:55
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 0
  - train gap: 2017-12-11T20:51:19:2017-12-11T21:35:56
  - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:18
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 1
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:56
  - validation gap: 2017-12-11T20:51:19:2017-12-11T20:58:46
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 2
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:56
  - validation gap: 2017-12-11T20:58:47:2017-12-11T21:06:12
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 3
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:56
  - validation gap: 2017-12-11T21:06:13:2017-12-11T21:13:38
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:56
  - validation gap: 2017-12-11T21:13:39:2017-12-11T21:21:04
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:56
  - validation gap: 2017-12-11T21:21:05:2017-12-11T21:28:30
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor I on fold 6
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:28:30
  - validation gap: 2017-12-11T21:28:31:2017-12-11T21:35:56
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 0
  - train gap: 2017-12-11T20:51:16:2017-12-11T21:35:39
  - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:15
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 1
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:39
  - validation gap: 2017-12-11T20:51:16:2017-12-11T20:58:39
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
```

```
Appliance: Double-pole Contactor II on fold 2
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:39
  - validation gap: 2017-12-11T20:58:40:2017-12-11T21:06:03
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 3
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:39
  - validation gap: 2017-12-11T21:06:04:2017-12-11T21:13:27
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:39
  - validation gap: 2017-12-11T21:13:28:2017-12-11T21:20:51
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:39
  - validation gap: 2017-12-11T21:20:52:2017-12-11T21:28:15
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Double-pole Contactor II on fold 6
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:28:15
  - validation gap: 2017-12-11T21:28:16:2017-12-11T21:35:39
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 0
  - train gap: 2017-12-11T20:51:19:2017-12-11T21:35:54
  - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:18
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 1
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:54
  - validation gap: 2017-12-11T20:51:19:2017-12-11T20:58:44
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 2
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:54
  - validation gap: 2017-12-11T20:58:45:2017-12-11T21:06:10
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 3
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:54
  - validation gap: 2017-12-11T21:06:11:2017-12-11T21:13:36
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:54
  - validation gap: 2017-12-11T21:13:37:2017-12-11T21:21:02
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:54
  - validation gap: 2017-12-11T21:21:03:2017-12-11T21:28:28
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan I on fold 6
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:28:28
  - validation gap: 2017-12-11T21:28:29:2017-12-11T21:35:54
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 0
  - train gap: 2017-12-11T20:51:14:2017-12-11T21:35:20
  - validation gap: 2017-12-11T20:43:52:2017-12-11T20:51:13
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 1
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:20
  - validation gap: 2017-12-11T20:51:14:2017-12-11T20:58:35
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 2
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:20
  - validation gap: 2017-12-11T20:58:36:2017-12-11T21:05:56
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 3
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:20
  - validation gap: 2017-12-11T21:05:57:2017-12-11T21:13:17
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 4
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:20
  - validation gap: 2017-12-11T21:13:18:2017-12-11T21:20:38
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 5
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:35:20
  - validation gap: 2017-12-11T21:20:39:2017-12-11T21:27:59
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Exhaust Fan II on fold 6
  - train gap: 2017-12-11T20:43:52:2017-12-11T21:27:59
  - validation gap: 2017-12-11T21:28:00:2017-12-11T21:35:20
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 0
  - train gap: 2018-02-19T22:54:21:2018-02-19T23:05:35
  - validation gap: 2018-02-19T22:52:28:2018-02-19T22:54:20
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 1
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:05:35
  - validation gap: 2018-02-19T22:54:21:2018-02-19T22:56:13
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 2
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:05:35
  - validation gap: 2018-02-19T22:56:14:2018-02-19T22:58:06
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 3
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:05:35
  - validation gap: 2018-02-19T22:58:07:2018-02-19T22:59:59
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 4
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:05:35
  - validation gap: 2018-02-19T23:00:00:2018-02-19T23:01:51
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 5
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:05:35
  - validation gap: 2018-02-19T23:01:52:2018-02-19T23:03:43
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine I on fold 6
  - train gap: 2018-02-19T22:52:28:2018-02-19T23:03:43
  - validation gap: 2018-02-19T23:03:44:2018-02-19T23:05:35
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 0
  - train gap: 2018-02-19T22:54:22:2018-02-19T23:05:37
  - validation gap: 2018-02-19T22:52:29:2018-02-19T22:54:21
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 1
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:05:37
  - validation gap: 2018-02-19T22:54:22:2018-02-19T22:56:14
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 2
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:05:37
  - validation gap: 2018-02-19T22:56:15:2018-02-19T22:58:07
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 3
```

```
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:05:37
  - validation gap: 2018-02-19T22:58:08:2018-02-19T23:00:00
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 4
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:05:37
  - validation gap: 2018-02-19T23:00:01:2018-02-19T23:01:53
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 5
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:05:37
  - validation gap: 2018-02-19T23:01:54:2018-02-19T23:03:45
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
Appliance: Milling Machine II on fold 6
  - train gap: 2018-02-19T22:52:29:2018-02-19T23:03:45
  - validation gap: 2018-02-19T23:03:46:2018-02-19T23:05:37
  - test gap: 2018-04-01 13:17:21+00:00 : 2018-04-03 15:48:47+00:00
==================Means and STD============
    "Pelletizer I": {
        "mean": [
            6739.594822767663,
            33.69756475642021
        ],
        "std": [
            51.17050304268481,
            23.82871894031847
        ]
    },
    "Pelletizer II": {
        "mean": [
            6740.168183499572,
            33.441182036036984
        ],
        "std": [
            51.36374825760136,
            23.88174982691653
        ]
    },
    "Double-pole Contactor I": {
        "mean": [
            6740.162164253307,
            33.446267619833215
        ],
        "std": [
            51.358292778354134,
            23.88155212711967
        ]
    },
    "Double-pole Contactor II": {
        "mean": [
            6740.3341079415095,
            33.42896490937573
        ],
        "std": [
            51.35046354326173,
            23.910003450832395
        ]
    },
    "Exhaust Fan I": {
        "mean": [
            6740.1638060896585,
            33.443767671038664
        ],
        "std": [
            51.36427418425424,
            23.88201536430618
        ]
    },
    "Exhaust Fan II": {
        "mean": [
            6739.9998979686925,
            33.59945378261505
        ],
        "std": [
            51.30972999555064,
            23.879451350676955
        ]
    },
    "Milling Machine I": {
        "mean": [
            6741.2522288269065,
            34.084129110700374
        ],
        "std": [
            50.04662160612179,
            22.93584371361278
        ]
    },
    "Milling Machine II": {
        "mean": [
            6741.299658712319,
            34.061662761187655
        ],
        "std": [
            50.04364615483314,
            22.930161463857242
        ]
    }
}
==================Scores on Test Set==================
==========Pelletizer I==========
F1 Score: mean:98.277985, std:0.050698
NDE: mean:0.023953, std:0.001329
SAE: mean:0.020778, std:0.003172
MSE: mean:91387035.324848, std:5071199.557461
MAE: mean:3015.271739, std:63.403069
====================
==========Pelletizer II==========
F1 Score: mean:97.426203, std:0.056077
NDE: mean:0.036899, std:0.000905
SAE: mean:0.037623, std:0.003866
MSE: mean:126397885.455240, std:3098862.570705
MAE: mean:4034.629246, std:46.871800
====================
==========Double-pole Contactor I==========
F1 Score: mean:97.165256, std:0.040858
NDE: mean:0.052699, std:0.004678
SAE: mean:0.072961, std:0.023182
MSE: mean:33946.848171, std:3013.086460
```

```
MAE: mean:79.755762, std:8.441514
====================
==========Double−pole Contactor II==========
F1 Score: mean:75.719913, std:0.046713
NDE: mean:0.682095, std:0.041950
SAE: mean:0.663258, std:0.048234
MSE: mean:305482.955720, std:18787.608793
MAE: mean:285.977601, std:13.561529
====================
==========Exhaust Fan I==========
F1 Score: mean:98.296734, std:0.026845
NDE: mean:0.048336, std:0.000579
SAE: mean:0.009735, std:0.006441
MSE: mean:313101.836834, std:3749.034802
MAE: mean:134.853799, std:1.384034
====================
==========Exhaust Fan II==========
F1 Score: mean:98.112254, std:0.308573
NDE: mean:0.049312, std:0.006959
SAE: mean:0.011347, std:0.005710
MSE: mean:915141.160972, std:129153.730853
MAE: mean:247.828238, std:27.127936
====================
==========Milling Machine I==========
F1 Score: mean:93.017162, std:0.884479
NDE: mean:0.110613, std:0.015437
SAE: mean:0.079064, std:0.047207
MSE: mean:86333084.235863, std:12048630.796999
MAE: mean:4317.012515, std:428.572732
====================
==========Milling Machine II==========
F1 Score: mean:93.771856, std:0.518441
NDE: mean:0.078719, std:0.004145
SAE: mean:0.060303, std:0.022049
MSE: mean:66722245.061129, std:3513088.121908
MAE: mean:3154.172028, std:159.196058
====================
==================ALL Scores on Test Set==================
==========Pelletizer I==========
————F1 Score————
        Fold 0:98.364813
        Fold 1:98.243309
        Fold 2:98.311877
        Fold 3:98.316503
        Fold 4:98.265508
        Fold 5:98.212417
        Fold 6:98.231469
————NDE————
        Fold 0:0.022815
        Fold 1:0.023567
        Fold 2:0.023336
        Fold 3:0.022635
        Fold 4:0.023685
        Fold 5:0.026762
        Fold 6:0.024869
————SAE————
        Fold 0:0.023007
        Fold 1:0.018743
        Fold 2:0.020942
        Fold 3:0.016217
        Fold 4:0.018874
        Fold 5:0.026848
        Fold 6:0.020815
————MSE————
        Fold 0:87045160.748986
        Fold 1:89915357.874667
        Fold 2:89032917.564497
        Fold 3:86358324.949354
        Fold 4:90366621.566992
        Fold 5:102105978.498630
        Fold 6:94884886.070812
————MAE————
        Fold 0:2947.199466
        Fold 1:3001.769835
        Fold 2:2973.522337
        Fold 3:2957.903211
        Fold 4:3013.048114
        Fold 5:3134.616171
```

```
    Fold 6:3078.843036
==========Pelletizer II==========
----------F1 Score----------
    Fold 0:97.474984
    Fold 1:97.463913
    Fold 2:97.481957
    Fold 3:97.308752
    Fold 4:97.411470
    Fold 5:97.443972
    Fold 6:97.398370
----------NDE----------
    Fold 0:0.036167
    Fold 1:0.036067
    Fold 2:0.036172
    Fold 3:0.038205
    Fold 4:0.037025
    Fold 5:0.038300
    Fold 6:0.036361
----------SAE----------
    Fold 0:0.040758
    Fold 1:0.038615
    Fold 2:0.035922
    Fold 3:0.034001
    Fold 4:0.035130
    Fold 5:0.045158
    Fold 6:0.033779
----------MSE----------
    Fold 0:123890018.191987
    Fold 1:123545697.080437
    Fold 2:123906166.793013
    Fold 3:130868654.244972
    Fold 4:126827860.781097
    Fold 5:131194594.916342
    Fold 6:124552206.178830
----------MAE----------
    Fold 0:4010.031500
    Fold 1:3982.504964
    Fold 2:3986.667176
    Fold 3:4098.552108
```

```
      Fold 4:4030.464167
      Fold 5:4109.194757
      Fold 6:4024.990048
==========Double-pole Contactor I==========
------F1 Score------
      Fold 0:97.196639
      Fold 1:97.146875
      Fold 2:97.161176
      Fold 3:97.212400
      Fold 4:97.212377
      Fold 5:97.130674
      Fold 6:97.096647
------NDE------
      Fold 0:0.048841
      Fold 1:0.050924
      Fold 2:0.052692
      Fold 3:0.048007
      Fold 4:0.056214
      Fold 5:0.049882
      Fold 6:0.062333
------SAE------
      Fold 0:0.058274
      Fold 1:0.058028
      Fold 2:0.062996
      Fold 3:0.052462
      Fold 4:0.096092
      Fold 5:0.063281
      Fold 6:0.119598
------MSE------
      Fold 0:31461.799130
      Fold 1:32803.471118
      Fold 2:33942.264968
      Fold 3:30924.551814
      Fold 4:36210.907507
      Fold 5:32132.093094
      Fold 6:40152.849570
------MAE------
      Fold 0:73.962445
      Fold 1:74.949926
```

```
        Fold 2:77.843094
        Fold 3:72.463883
        Fold 4:87.544691
        Fold 5:74.527468
        Fold 6:96.998825
============Double-pole Contactor II============
--------F1 Score--------
        Fold 0:75.803204
        Fold 1:75.699780
        Fold 2:75.708381
        Fold 3:75.728771
        Fold 4:75.758456
        Fold 5:75.696205
        Fold 6:75.644595
--------NDE--------
        Fold 0:0.665503
        Fold 1:0.636510
        Fold 2:0.708206
        Fold 3:0.766733
        Fold 4:0.670684
        Fold 5:0.638028
        Fold 6:0.688998
--------SAE--------
        Fold 0:0.648095
        Fold 1:0.608482
        Fold 2:0.693342
        Fold 3:0.758896
        Fold 4:0.654190
        Fold 5:0.609650
        Fold 6:0.670152
--------MSE--------
        Fold 0:298052.304643
        Fold 1:285067.550158
        Fold 2:317177.257856
        Fold 3:343389.174538
        Fold 4:300372.658141
        Fold 5:285747.088198
        Fold 6:308574.656508
--------MAE--------
```

```
        Fold 0:280.067812
        Fold 1:272.656938
        Fold 2:293.251724
        Fold 3:314.923788
        Fold 4:281.433690
        Fold 5:273.191576
        Fold 6:286.317681
==========Exhaust Fan I==========
--------F1 Score--------
        Fold 0:98.255080
        Fold 1:98.292897
        Fold 2:98.300477
        Fold 3:98.319429
        Fold 4:98.308748
        Fold 5:98.336285
        Fold 6:98.264220
--------NDE--------
        Fold 0:0.049344
        Fold 1:0.047993
        Fold 2:0.047528
        Fold 3:0.047838
        Fold 4:0.048409
        Fold 5:0.048353
        Fold 6:0.048887
--------SAE--------
        Fold 0:0.022423
        Fold 1:0.005505
        Fold 2:0.004332
        Fold 3:0.007877
        Fold 4:0.008620
        Fold 5:0.003464
        Fold 6:0.015928
--------MSE--------
        Fold 0:319633.826175
        Fold 1:310879.886210
        Fold 2:307868.357065
        Fold 3:309873.487466
        Fold 4:313575.610831
        Fold 5:313211.254058
        Fold 6:316670.436036
————MAE————
        Fold 0:137.355420
        Fold 1:135.113573
        Fold 2:133.906015
        Fold 3:133.162215
        Fold 4:133.587535
        Fold 5:134.696039
        Fold 6:136.155798
==========Exhaust Fan II==========
————F1 Score————
        Fold 0:98.226027
        Fold 1:98.239368
        Fold 2:97.357728
        Fold 3:98.236218
        Fold 4:98.249196
        Fold 5:98.270640
        Fold 6:98.206604
————NDE————
        Fold 0:0.046634
        Fold 1:0.046366
        Fold 2:0.066322
        Fold 3:0.046208
        Fold 4:0.046269
        Fold 5:0.045911
        Fold 6:0.047476
————SAE————
        Fold 0:0.003079
        Fold 1:0.011479
        Fold 2:0.023522
        Fold 3:0.009916
        Fold 4:0.008283
        Fold 5:0.011383
        Fold 6:0.011770
————MSE————
        Fold 0:865438.408164
        Fold 1:860458.880435
        Fold 2:1230814.334351
        Fold 3:857528.585428
        Fold 4:858657.214349
        Fold 5:852019.768703
        Fold 6:881070.935374
----------MAE----------
        Fold 0:243.893851
        Fold 1:232.808257
        Fold 2:313.548953
        Fold 3:232.963289
        Fold 4:238.668617
        Fold 5:232.734240
        Fold 6:240.180457
==========Milling Machine I==========
----------F1 Score----------
        Fold 0:93.344865
        Fold 1:93.418797
        Fold 2:93.913605
        Fold 3:90.970136
        Fold 4:93.166269
        Fold 5:92.875529
        Fold 6:93.430934
----------NDE----------
        Fold 0:0.105497
        Fold 1:0.110912
        Fold 2:0.104278
        Fold 3:0.147189
        Fold 4:0.096568
        Fold 5:0.105897
        Fold 6:0.103949
----------SAE----------
        Fold 0:0.066685
        Fold 1:0.076103
        Fold 2:0.089949
        Fold 3:0.186421
        Fold 4:0.033569
        Fold 5:0.044204
        Fold 6:0.056519
----------MSE----------
        Fold 0:82339968.525655
        Fold 1:86566334.539304
        Fold 2:81389115.802078
        Fold 3:114880905.649382
        Fold 4:75371106.664862
        Fold 5:82652413.161698
        Fold 6:81131745.308059
————MAE————
        Fold 0:4154.214175
        Fold 1:4230.803073
        Fold 2:4164.139675
        Fold 3:5349.229647
        Fold 4:3966.167607
        Fold 5:4197.197773
        Fold 6:4157.335652
==========Milling Machine II==========
————F1 Score————
        Fold 0:94.088053
        Fold 1:93.812420
        Fold 2:94.140369
        Fold 3:93.824317
        Fold 4:92.605349
        Fold 5:94.286381
        Fold 6:93.646102
————NDE————
        Fold 0:0.075681
        Fold 1:0.085463
        Fold 2:0.076664
        Fold 3:0.073826
        Fold 4:0.079735
        Fold 5:0.075716
        Fold 6:0.083946
————SAE————
        Fold 0:0.053724
        Fold 1:0.085843
        Fold 2:0.069798
        Fold 3:0.041024
        Fold 4:0.020336
        Fold 5:0.066366
        Fold 6:0.085032
————MSE————
```

```
    Fold 0:64147592.007647
    Fold 1:72438926.845482
    Fold 2:64980608.183700
    Fold 3:62575235.962060
    Fold 4:67583608.267814
    Fold 5:64176806.237862
    Fold 6:71152937.923341
————MAE————
    Fold 0:2999.143837
    Fold 1:3218.910360
    Fold 2:3071.122201
    Fold 3:3026.586988
    Fold 4:3295.295094
    Fold 5:3018.063967
    Fold 6:3450.081748
```