Programme Area: Smart Systems and Heat

Project: WP1 Appliance Disaggregation

Title: Data Analysis Pre-processing

Abstract:

The purpose of this deliverable is to explain the process of reading the multi-source raw data from the hard disks and extracting features that are useful for machine learning algorithms. The process of synchronisation of the electricity, water and HEMS data is presented. A brief description of the code developed to achieve the new data representation is provided.

Context:

The High Frequency Appliance Disaggregation Analysis (HFADA) project builds upon work undertaken in the Smart Systems and Heat (SSH) programme delivered by the Energy Systems Catapult for the ETI, to refine intelligence and gain detailed smart home energy data. The project analysed in depth data from five homes that trialed the SSH programme's Home Energy Management System (HEMS) to identify which appliances are present within a building and when they are in operation. The main goal of the HFADA project was to detect human behaviour patterns in order to forecast the home energy needs of people in the future. In particular the project delivered a detailed set of data mining algorithms to help identify patterns of building occupancy and energy use within domestic homes from water, gas and electricity data.

Disclaimer: The Energy Technologies Institute is making this document available to use under the Energy Technologies Institute Open Licence for Materials. Please refer to the Energy Technologies Institute website for the terms and conditions of this licence. The Information is licensed 'as is' and the Energy Technologies Institute excludes all representations, warranties, obligations and liabilities in relation to the Information to the maximum extent permitted by law. The Energy Technologies Institute is not liable for any errors or omissions in the Information and shall not be liable for any loss, injury or damage of any kind caused by its use. This exclusion of liability includes, but is not limited to, any direct, indirect, special, incidental, consequential, punitive, or exemplary damages in each case such as loss of revenue, data, anticipated profits, and lost business. The Energy Technologies Institute does not guarantee the continued supply of the Information. Notwithstanding any statement to the contrary contained on the face of this document, the Energy Technologies Institute confirms that it has the right to publish this document.

Bournemouth University

Project: HFADA

HIGH FREQUENCY APPLIANCE DISAGGREGATION ANALYSIS

© 2017 Bournemouth University. The information in this document is the property of Bournemouth University and may not be copied or communicated to a third party or used for any purpose other than that for which it is supplied without the express written consent of Bournemouth University.

Contents

1. History

Date	Issue	Details of Change
20/11/2017	Version 1.0	First Version. Authors: Saad Mohamad Chemseddine Mansouri Hamid Bouchachia
06/12/2017	Version 2.0	Second Version Same authors

2. Documents Referenced

Ref	Document	Title
1	Word document that describes the HEMS data.	Data collection and data format – ELECTRIC, WATER and HEMS-V1 MONITORING
	Word document that describes the HEMS V1 Mongo data base structure.	HEMS V1 Mongo Data Base Structure

3. Glossary of Terms

Ref	Description
ETI	Energy Technologies Institute
HEMS	Home Energy Management System (also referred to as HEMS V1)
HFAD	High Frequency Appliance Detection
NTP	Network Timing Protocol
RMS	Root Mean Square

4. Executive Summary

The purpose of this deliverable is to explain the process of reading the multi-source raw data from the hard disks and extracting features that are useful for machine learning algorithms. The process of synchronisation of the electricity, water and HEMS data is presented. Moreover, a brief description of the code developed to achieve the new data representation is given.

1. Introduction

The basic aim of this project is to develop pattern mining and recognition using information from multiple utilities (electricity signals: current and voltage, water flow, gas flow, other sensory measurements and input) that allow to identify useful occupancy and energy usage patterns. To achieve this goal, data has to be pre-processed and prepared to be fully exploited. Three main steps are necessary: data verification, cleansing, and feature extraction. Before delving into the details, we highlight the structure of this report. Section 2 discusses the preprocessing steps which mainly cover data preparation, data reading and data synchronisation and alignment. Section 3 summarises the outcome of the pre-processing steps. Section 4 outlines the code structure and its execution. Section 5 concludes this report.

2. Preprocessing

Data has been provided over many hard drives in different formats. As outlined earlier, three main data sources will be pre-processed: Water data, Electricity data and HEMS data[1]. Water data is stored in text files with sampling rate of 10 seconds and is synchronised to NTP approximately once per month. Electricity data is stored in wave files with sampling rate of 4.88 µs and is synchronised to NTP every 28min 28sec. HEMS data is stored in a Mongo database with sampling rates differing according to the type of the data and sensors generating it. In this deliverable, we implement a Python code that reads the data from these different sources, synchronises its timestamps to NTP timestamps, extracts features and aligns the data samples to one timestamp by measurement. Figure 1 presents the steps of data pre-processing.

2.1 Data Preparation Steps

To prepare data for mining and knowledge discovery, a number of pre-processing steps are often necessary.

a) Data verification

While the electricity and water data looks quite consistent, HEMS data seems to contain some inconsistencies. These have been raised for further investigation by ESC and ETI. The following highlights the outstanding issues:

- Why do we have temperature in the WIRELESS_RADIATOR_VALVE network?
- Why do we have 55 and 99 values of the valves position? The manual states that the value should be -1 0 or 1. (Open, in between and close)?
- Why does “NetworkId” of hot and cold-water temperature measure humidity?
- Why does RadiatorBox measure humidity?
- Is there more than one boiler in the house? The firing data includes an array of two dictionaries!
- The discrepancy of the time between in and out radiator water temperature measurements is high, sometime more than 120 second, why?

[1] Gas data and sensor information. Although exploiting this information could be done in the fourth deliverable (D4), we may use this information in the second deliverable (D2). Other additional information about the existing appliances, the topology of building, etc. will be exploited in D4. Note that this additional information is not considered in the pre-processing stage.

- There is a mistake in the database schema; home_id and sensor_data are not connected. Home_id ' role_id is the hub number. There are 5 hubs with the following IDS: 11 18 25 32 39 and 5 houses with the following IDS: 10 17 24 31 38. Can we assume that hub 11 is for house 10, 18 for house 17 and so on?
- The recording time of different data sources is not the same and the HEMS data does not cover all houses.

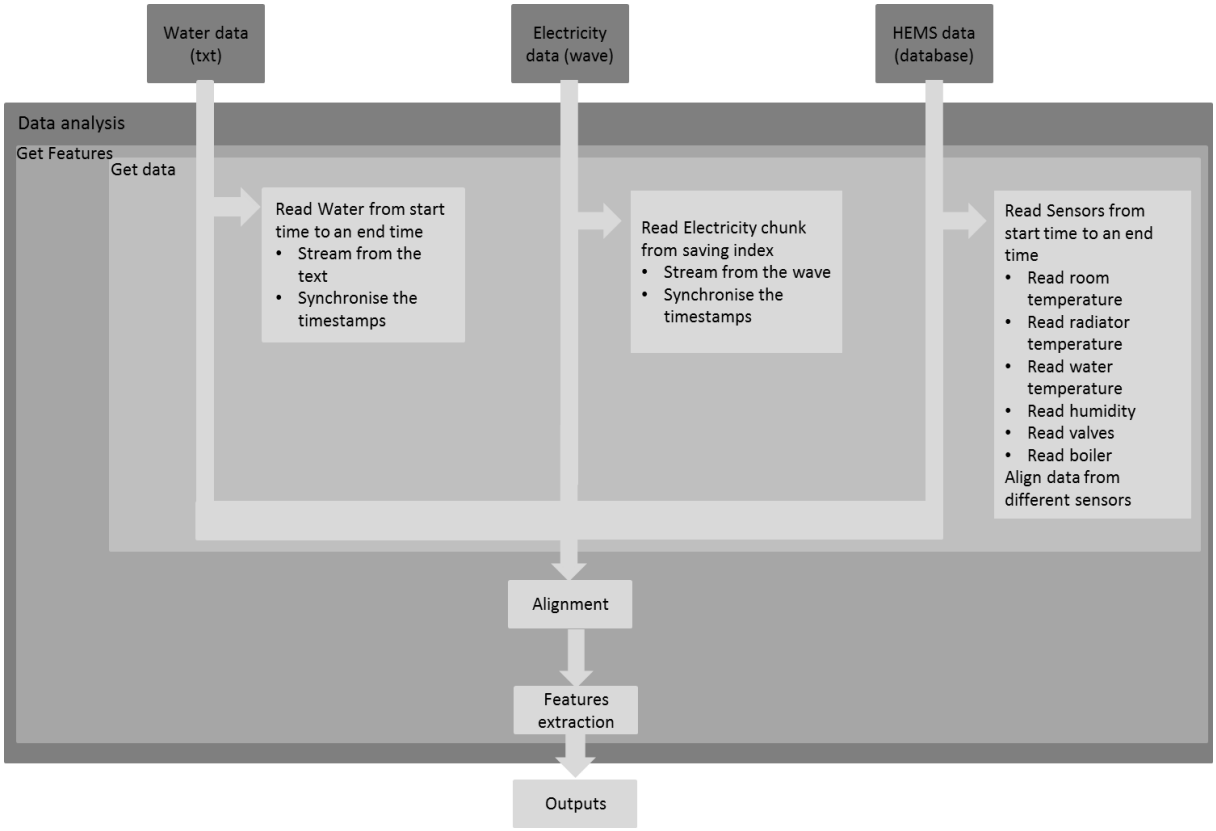


Figure 1: Preprocessing steps

b) Data clean-up

Luckily data seems clean; no missing values have been observed with the first hard drives.

c) Feature extraction

Features extraction can be seen as a form of data compression. It helps reduce unnecessary redundancies in the raw data signal. Pushed by the complexity of the mining task and motivated by the informativeness and simplicity of the water and sensors data, we only, at this stage, extract few features form the electricity data:

- Real and reactive power: these features have been shown to be very useful (alone or accompanied with other features) in many conventional non-intrusive load monitoring approaches [1-7]. The importance of these features or features derived from them is that they convey information about the load of the appliance as well as the nature of it (difference between reactive and active power). Although in this work, our main interest is not appliance recognition, features help distinguish between different appliances would be informative and useful in the pattern mining task. Another advantage of these features is that they do not require high sampling rate and therefore expensive current

and voltage meters. However, the provided electricity data by ETI is sampled at somewhat high rate. To exploit the information offered with the high sampling rate. We also extract frequency domain features.

- Spectrum power: This feature provides information about the waveforms. Different waveforms can characterise different types of appliances. Hence, it is expected that these features will be very useful in the mining task. Adopting these features was also inspired by the work of the researchers at MIT as well as other research studies [8-13]. The harmonics of the signal can also uniquely characterise non-linear loads that draw non-sinusoidal current during the operation. They have been used in combination with real and reactive power features [14, 15]. In this work, we compute the RMS spectrum power over fixed ranges of frequencies. The size and number of these ranges are provided as parameter of the electricity extraction function. In Section 5, we show figures of these features for different current waveforms sampled from the ETI electricity data.

The water and gas datasets retain their original representation.

2.2 Reading the Data

As shown in Fig. 1, the implemented code[2], written in Python, provides two main functions:

- ***Get_data*:** this function takes as parameters data source (0: electricity, 1: water, 2: HEMS), the start time of reading and the end time of reading. For water data, it streams the data in strings from the text files; converts the read timestamps and the measured water flows into numbers and save them in global Python's NumPy matrix. A synchronisation function is then called. It synchronises the PC clock timestamps of samples within each month to NTP timestamp. The synchronisation is done as follows:

$$timestampNTP(i) = timestampclock(i) + i * \frac{totalTimeShiftDuringOnMonth}{numberOfSampleDuringTheMonth}$$

In this equation, we assume that the total shift (between NTP and PC clock) can be distributed over the samples. The Numpy matrix of the old timestamps is then updated by the modified timestamps. Finally, the function returns a matrix containing the data collected between the start and end time given as input to the function.

Electricity data is read in chunks starting from the given start time and chunk size which are given as input. Because the size of electricity data is huge, the chunk size should be smaller than 28minutes 28seconds. Note that the chunk size is given as global parameter for the class constructor. This function streams from wave files, computes the timestamps that should be associated with each sample using the given sampling rate and stores the data in Python's Numpy matrix. A synchronisation function is then called to obtain the NTP timestamp:

$$timestampNTP(i) = timestampclock(i) + i * \frac{totalTimeShiftDuring28minutes28seconds}{numberOfSampleDuring28minutes28seconds}$$

[2] The Python class consists of two main public functions.

The stored Numpy matrix gets updated by the modified timestamp. Finally, the function returns a matrix containing the data collected from the start time and with a given chunk size.

To collect data from HEMS, pymongo is used to query mongodb database in which the data is stored. Details on how to set-up pymongo, import the database and index it are provided in Section 5. Sensor data is read according to the start and end time given as parameters to the function. As shown in Fig.1, each data type is read using different sub-functions. This data is then aligned as shown in Fig. 2. The union of all data samples timestamps is stored in one matrix. Each row of this matrix includes a timestamp and the corresponding values of the sensors. If for some sensors, there are no measurements taken at the timestamp, the values measured at the previous timestamp are taken. The *get data* function then returns one matrix that has all data samples timestamps and all sensors measurements.

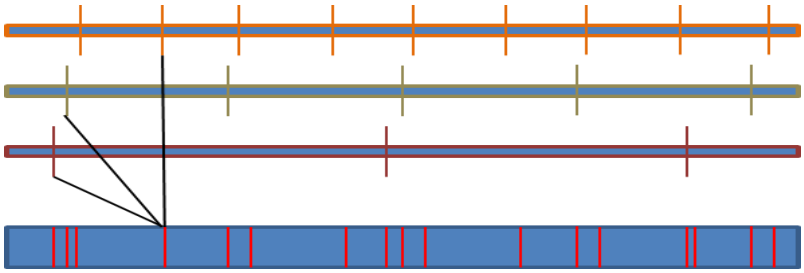


Figure 2: Alignment of the data

- ***Get_features*:** this function takes the time granularity as a parameter. It uses the *get data* function to read water, electricity and sensor data. The data chunks are then split into batches according to the given time granularity. From electricity batches, we extract different features that will be used in the mining process. Details about the extracted features are given in the next section. Features might also be extracted from water data as well as the sensors data. The extracted features are then aligned following the same philosophy as the sensors data alignment. Finally, the function returns, in matrix form, the chunk of the features extracted along with the time stamps of each row of features. Table 1 shows the format of the output matrix.

Table 1: Extracted features (strcuture of the formatted data)

timestampNTP	Water	Electricity			HEMS						
		Real power	Reactive power	RMS Spectrum power over different frequency ranges	Gas meter	Temperature			humidity	Radiators' valve	Firing boiler
						Rooms	radiators	water			
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.			.	.	.	.	.	.
.	.	.	.			.	.	.	.	.	.

3. Outcome of the Preprocessing

To be able to feed this data to machine learning algorithms, the multi-source data has been transformed into a tabular form (i.e., in a form of streaming vectors or batch of vectors -

matrices). These vectors are aligned as our mining algorithm will not support asynchronous learning; though, this can be done, it is not part of the proposal. Hence, synchronisation and alignment have been implemented. Furthermore, the huge size of the electricity data caused by the high frequency rate makes it hard for the memory to accommodate the whole dataset. To cope with this, we (1) divided the data into chunks and (2) extracted selected and informative features form the electricity raw input to reduce the redundancies.

Table 1 shows the format of the output batch of features. All features are aligned to the same NTP timestamps. The feature of the water data is the water flow itself. We could, however, take the variation of the water flow instead, by adding one line to the existing function. From the electricity data, real power, reactive power and the Root-Mean-Square of spectrum power over fixed frequency ranges are considered as features. The sensor data is stored in the output matrix in the following order: data from gas meter, temperature of rooms, radiators and water, relative humidity of room, the state of the radiators‘ valve and the state of the boiler firing switch. We do not extract features from these measurements. Although, the raw input of these measurements provides good information, taking the variation of these measurements as features can be done.

4. Python Code Related to the Deliverable

As part of this deliverable, we submit the Python code which includes the module explained above as well as a module that allows saving the extracted features or raw data in CSV files. Note that these codes are for research purpose (as stated in the proposal) and are specific for the data at hand. In the following we provide details about the code and how to run it.

4.1 Requirements

- Python (preferably Python 2.7.12) should be installed.
- The following packages are used: scipy, glob, numpy, datetime, pymongo, time, re, sys and csv
- install MongoDB
- Import roledata.bson file to mongodb. The data should be imported with database name: “db_name” and collection name: “collection_name”
- Create the following indices for mongo db:

```
[ {       "v" : 1,
        "key" : {
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.humidity.timestamp" : 1
        },
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.humidity.timestamp_1",
```

```
        "ns" : "db_name.collection_name"
    },
    {
        "v" : 1,
        "key" : {
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.switch_variable.timestamp" : 1
        },
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.switch_variable.timestamp_1",
        "ns" : "db_name.collection_name"
    },
    {
        "v" : 1,
        "key" : {
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.energy_gas.timestamp" : 1
        },
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.energy_gas.timestamp_1",
        "ns" : "db_name.collection_name"
    },
    {
        "v" : 1,
        "key" : {
```

```
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.temperature.0.timestamp" : 1
        },
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.temperature.0.timestamp_1",
        "ns" : "db_name.collection_name"
    },
    {
        "v" : 1,
        "key" : {
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.temperature.1.timestamp" : 1
        },
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.temperature.1.timestamp_1",
        "ns" : "db_name.collection_name"
    },
    {
        "v" : 1,
        "key" : {
            "type_token" : 1,
            "role_id" : 1,
            "value.networkId" : 1,
            "value.data.switch_onoff.0.timestamp" : 1
        },
```

```
        "name" :
"type_token_1_role_id_1_value.networkId_1_value.data.switch_onoff.0.timestamp_1",

        "ns" : "db_name.collection_name"

    }

]
```

4.2 Settings

The main class is called *signals*, it takes as input the following:

- The path of the electricity wave files directory
- The path of the water txt file
- The shifting time of the water data
- The starting time of the data reading
- The home ID
- The size of batches each wave file will be spited into. This is 7 by default.

4.3 Running the Code

- Plug in the hard drive
- Initiate the class with the right input.
- Two public functions, described in the report are to be used: get_features and get_data.
- Once, the data (water or electricity) on the hard drive has been totally read, the user will be requested to enter a new electricity or/and water data paths. Note that it is the user responsibility to make sure that the right consecutive hard drive that follows exactly from the read one is plugged.

Save_in_CSV can be used to save the data or features in CSV files. Here is an example how saving in a CSV file can be done:

python Save_in_CSV.py /media/dung/ETI1144/ProjectPico/Water/20063557_0.txt /media/dung/ETI1144/ProjectPico/T01 10 47 3 2017-04-10T14:40:33

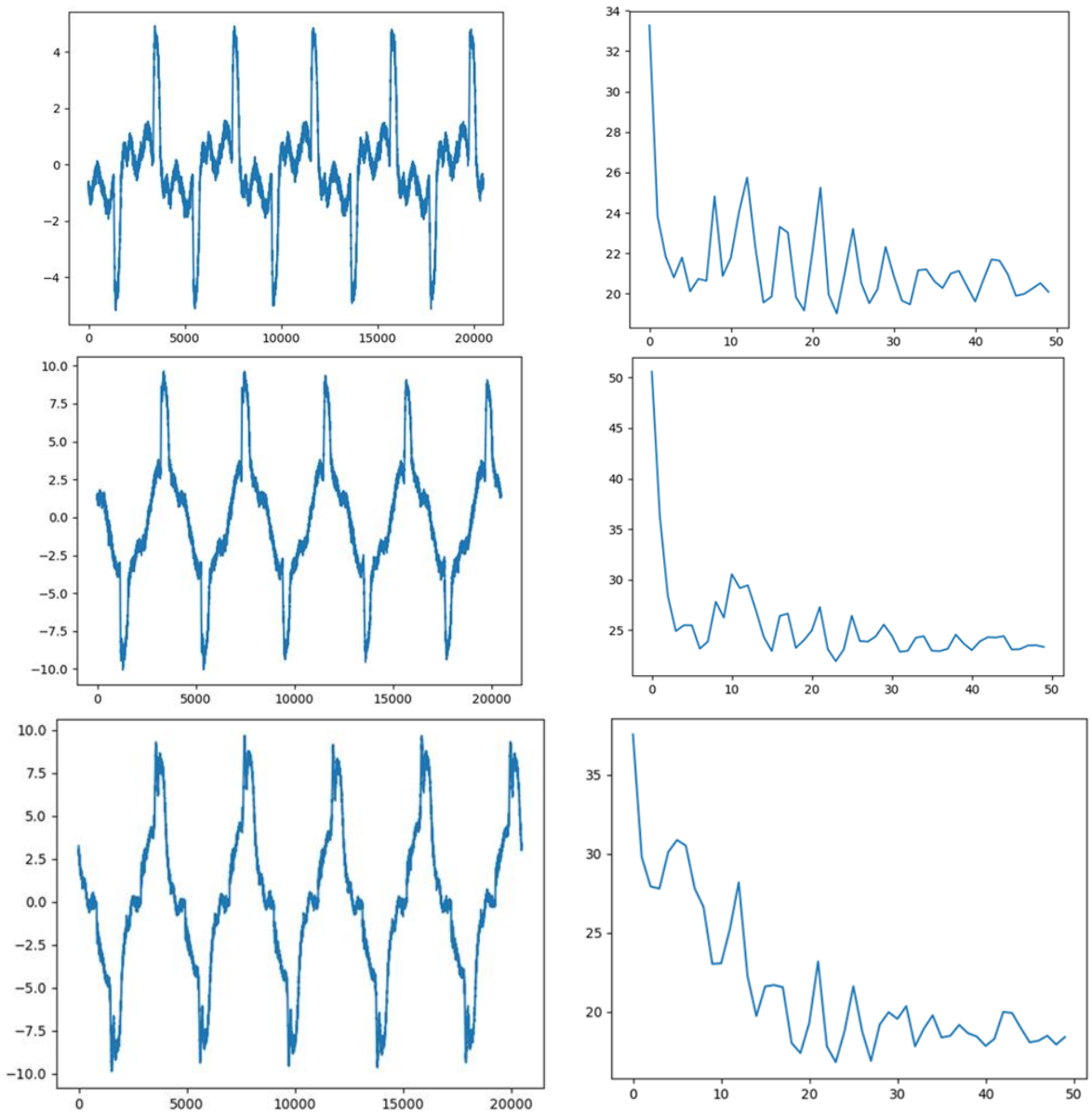
Details about the given parameters and other options can be found as comments in the code “Save_in_CSV”

4.4 Sample of Extracted Features

A CSV file of features extracted with 10 second granularity is provided with the code. Below are few figures to show the outcome of the pre-processing. Each couple of figures represents the current signal and the extracted spectrum power over 50 frequency ranges. These figures represent measurements taken at different times (around 1 hour between the data chunks plotted). Note that the samples shown are not in their original scale. It can be seen clearly that different waveforms are associated with different spectrum power. Even small changes in the waveforms results lead to different power spectrum (Fig. 3).

5. Conclusion

In this report, the goals and details of the pre-processing steps are presented. The format of the output is described. We motivate the purpose of the pre-processing steps and explain why it is a key requirement for the next stages of the project. Finally, the requirements and settings of the Python codes are presented and how the code can be run.



(a): current waveforms

(b): RMS spectrum power over fixed ranges of frequencies

Figure 3: Feature extraction

6. References

[1] Hart, George William. "Nonintrusive appliance load monitoring." Proceedings of the IEEE 80.12 (1992): 1870-1891.

[2] Marchiori, Alan, et al. "Circuit-level load monitoring for household energy management." IEEE Pervasive Computing 10.1 (2011): 40-48.

[3] Norford, Leslie K., and Steven B. Leeb. "Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms." Energy and Buildings 24.1 (1996): 51-64.

[4] Farinaccio, Linda, and Radu Zmeureanu. "Using a pattern recognition approach to disaggregate the total electricity consumption in a house into the major end-uses." *Energy and Buildings* 30.3 (1999): 245-259.

[5] Marceau, Medgar Louis, and R. Zmeureanu. "Nonintrusive load disaggregation computer program to estimate the energy consumption of major end uses in residential buildings." Energy conversion and management 41.13 (2000): 1389-1403.

[6] Figueiredo, Marisa, Ana De Almeida, and Bernardete Ribeiro. "Home electrical signal disaggregation for non-intrusive load monitoring (NILM) systems." Neurocomputing 96 (2012): 66-73.

[7] Belley, Corinne, et al. "An efficient and inexpensive method for activity recognition within a smart home based on load signatures of appliances." Pervasive and Mobile Computing 12 (2014): 58-78.

[8] Laughman, Christopher, et al. "Power signature analysis." IEEE power and energy magazine 99.2 (2003): 56-63.

[9] Lee, Kwangduk D., et al. "Estimation of variable-speed-drive power consumption from harmonic content." IEEE Transactions on Energy Conversion 20.3 (2005): 566-574.

[10] Wichakool, Warit, et al. "Modeling and estimating current harmonics of variable electronic loads." *IEEE Transactions on power electronics* 24.12 (2009): 2803-2811.

[11] Srinivasan, D., W. S. Ng, and A. C. Liew. "Neural-network-based signature recognition for harmonic source identification." IEEE Transactions on Power Delivery 21.1 (2006): 398-405.

[12] Ruzzelli, Antonio G., et al. "Real-time recognition and profiling of appliances through a single electricity sensor." *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*. IEEE, 2010.

[13] Li, Jiaming, Sam West, and Glenn Platt. "Power decomposition based on SVM regression." Modelling, Identification & Control (ICMIC), 2012 Proceedings of International Conference on. IEEE, 2012

[14] Najmeddine, Hala, et al. "State of art on load monitoring methods." Power and Energy Conference, 2008. PECon 2008. IEEE 2nd International. IEEE, 2008.

[15] Srinivasan, D., W. S. Ng, and A. C. Liew. "Neural-network-based signature recognition for harmonic source identification." IEEE Transactions on Power Delivery 21.1 (2006): 398-405.