Using a Cascading Classifier to improve the Recognition Performance of NIALM

Marcel Mathis, Alexander Klapproth

CC-iHomeLab, Lucerne University of Applied Sciences And Arts

Abstract

People are willing to save energy yet most have little to no understanding about the energy consumption of their household devices. Non-intrusive appliance load monitoring (NIALM) is an approach to automatically measure a household's energy consumption and identify switched-on appliances. This paper presents our ongoing research on improving electric load recognition thus helping people to better understand their energy consumption.

Depending on what current NIALM approaches identify, they have either high recognition accuracy or can recognize a lot of different devices. This approach uses a two stage cascading algorithm which recognizes groups of similar devices states in the first and individual devices states in the second stage. From a user perspective, the first classifier is an out-of-the-box algorithm with high recognition accuracy due to a low number of classes. The second classifier is trained by users with specific devices they want detailed information about. Our novel approach has an overall recognition accuracy of 87% and more.

Keywords—NIALM; NILM; load disaggregation; cascade classifier; device categorization; device labelling; load recognition; machine learning algorithm; BrTree; class weighting;

Introduction

While low power consumption of electrical devices is of growing importance for users and manufactures resulting in more power efficient devices, the overall consumption is growing. In Switzerland, one reason for the increasing consumption is the growing population. Another reason is the raising number of electrical household appliances. The Swiss government published different strategies describing how the total energy consumption can be decreased in homes [1], e.g.:

a) Substitution of devices with more efficient replacements
b) Automatic switch off of unused devices
c) Visualization of energy consumption to give users a better understanding of their energy consumption

It is getting more and more difficult to identify household devices that significantly contribute to the energy consumption. New devices have many device states (e.g. a blender can have different spin levels influencing its energy consumption) and it is not straight forward to know the associated energy consumption. In addition, most users have little to no understanding about their energy consumption although they are willing to save energy.

Non-intrusive appliance load monitoring (NIALM) is one approach to face these challenges: Based on low cost measurement systems or smart meter values, NIALM automatically detects and classifies switched-on appliances. Thus, NIALM automatically assess how much energy each appliance consumes and opens new possibilities to save energy by better informing the user: Inefficient devices can be easily identified, even automatically switched off, and real time information on consumption can be provided to users to increase their energy awareness at home.

One of the major problems of NIALM is adequate recognition accuracy. According to Zeifmann [2], accuracy above 80% is required to gain user acceptance. Mathis et al. [3] showed that constant recognition accuracy has to be achieved in spite of the raising number of electrical appliances. They introduce a labelling method device groups get recognized with high accuracy. Thus, NIALM applications necessitate a trade-off between the recognition of a large number of devices versus good recognition accuracy. As fewer classes have to be recognized as more accurate a classifier gets. For example, if there are only two devices and the algorithm always recognizing the same device, results in accuracy of 50%.

To reduce the number of devices Mathis et al. introduced a different labelling method with a low number of device categories [3]. They achieved constantly good recognition accuracy. The algorithms of this method can be easily trained by the manufacturer. The disadvantage is that information about individual devices cannot be determined by their system. In this paper, we show an extension to that method which also delivers detailed information about specific devices. We propose a two-stage cascaded classifier that recognizes the device category first and the specific device second.

Related Work

A NIALM system was first proposed by Hart [4] more than 20 years ago. A good overview on NIALM research since then was compiled by Liang [5], [6]. He explains the main challenge in NIALM to reliably detect different working states of electrical devices in order to maximize the device recognition accuracy. Usually, researchers dealing with measurements based on macro level group devices and label these groups (e.g. [7], [8]), whereas researchers that use micro level data label each device. Micro level denotes sampling rates higher than the fundamental waveform of the AC signal whereas macro level denotes lower rates. To our knowledge, there is no prior work combining device group and device labelling approaches. The two stage cascade classifier described in this paper relies on both: the first stage involves the device group and the second the device state labelling approach.

This work builds on Mathis et al.'s work [3] on labelling device states to categories to maximize recognition accuracy. Further Mathis et al.'s [9] algorithm has been used in the first stage.

Methods

The first stage of the cascade algorithm detects events and classifies their device class. This stage is intended to be trained by a NIALM manufacturer and to work out-of-the-box from a user perspective. The goal of this stage is to reliably detect events and to provide a few basic functionalities: It will be able to inform users whether they spend most money of the energy bill on lighting, electronic devices or on other device categories. Additionally, information about the total energy consumption and the amount of energy used for devices that are switched on 24h a day is provided. The algorithm classifies all samples according to their normed current waveform (NL Label), see [3] for details. For the recognition and event detection the BFTree algorithm [10] is used [9]. This algorithm periodically classifies the feature vector $\overrightarrow{F_P}$. For every event this input feature vector $\overrightarrow{F_P}$ is sent to the second stage as an event based feature vector $\overrightarrow{F_E}$.

The second stage disaggregates individual device states that are trained by NIALM users. From the user's perspective the most interesting devices should be visualized with the best possible accuracy and as much information as possible within this stage. Improved recognition accuracy is expected as the algorithm must only analyse a selection of devices instead of all devices or device states. This approach also heavily decreases the complexity of the algorithm. The algorithm of the second stage labels the device measurements according to their device state (DSL Label).

An overview of the two stage cascading algorithm is given in Figure 1.

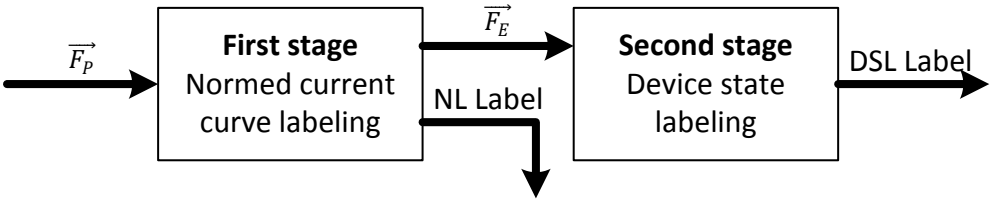

Figure 1 Schema of the two stage cascading NIALM algorithm. The first stage detects events which are then further analysed by the second stage to obtain device specific information.

Periodically feature vector $\overrightarrow{F_P}$

The quality of the input data for the cascade algorithm is of critical significance. First the voltage and current are measured at 5 kHz sampling rate. To keep the complexity of the algorithms low the delta curve is then calculated. The delta curve is defined as the difference between two measurements under the assumption that only one device changes its state between the measurements. The exact procedure is explained in [9].

From the delta curve, a feature vector is calculated. These vectors are used to decrease the amount of data and to extract features with high information content for the recognition task. Different input feature vectors have been compared in [3]. The best results of our algorithm in the first stage are achieved with a feature vector containing absolute Fourier Transform values. For this all odd harmonics up to the 11th order and the complex power values were used as input feature vector. The input feature vector $\overrightarrow{F_P}$ therefore consists of eight elements and is every second calculated.

First stage classifier

In the first stage of the algorithm, a BFTree [10] classifies the input vector $\overrightarrow{F_P}$ every second on a dedicated system. That means users can have a single NIALM system at meter level or they can have more of them on specific parts of the household to improve the recognition accuracy. Each device belongs to a device category (NL Label) [3]. If the user buys a new device, there is no retraining necessary as all available devices belong to such a category. The algorithm can be trained by the manufacturer and works from a user perspective out-of-the-box. The used algorithm of this first stage was developed in [9]. There events are measurements not classified as noise. Such events are forwarded to our second stage. The new feature vector $\overrightarrow{F_E}$ is identical to $\overrightarrow{F_P}$ but occurs only on events instead of every second. A reason for the event based approach of $\overrightarrow{F_E}$ is the usually low communication bandwidth between the measurement unit and the central unit. A central unit is seen as a device that calculates trends, histories and other consumption details for visualization purposes. It can be located in the household or could be implemented as a web service. The dedicated system runs on a low power system.

A BFTree is used for the first stage classification and requires little computation thus little energy to run. A BFTree is a binary tree with logarithmic complexity in memory-use and computation. Low energy consumption of the NIALM system itself is important, because NIALM is a way to cut down the total energy consumption of a household.

The following setup taken from [9] is used to achieve optimal results: Noise is trained with white Gaussian noise with 80 mA variance and the measurements of the training devices are overlaid with the same kind of noise. This large amount of noise is necessary to avoid algorithm over-fitting.

Second stage classifier

If the first stage detects an event, the periodic feature vector $\overrightarrow{F_P}$ is forwarded as an event based vector $\overrightarrow{F_E}$ to the 2nd stage. The NL label (i.e. the recognized device class) from the first stage is unknown to avoid error propagation. The goal of the 2nd stage is to provide more energy consumption details of preselected devices. If the user is for example interested in the energy consumption of his new computer monitor he can train that device. From then he will get a detailed energy consumption history of this device without buying a separate measurement system. An example of the algorithms

work is provided in Figure 2: The algorithm has exclusively trained to recognize a monitor. Thus, in this example other devices' energy consumption is of no interest to the user. The algorithm has to classify each event either as state change of the monitor or as unknown. Noise also belongs to the class unknown devices.

The biggest challenge is to differentiate between known and unknown devices. Figure 2 shows an example of two different devices. While the devices serve different purposes they have both similar power values and also a similar phase shift. If only the monitor class is known to the algorithm, the light bulb is probably misclassified as a monitor rather than an unknown device. Misclassifications mean that the user receives incorrect information and thus to false conclusions on how to save energy. Eventually users will lose confidence in the system. The more devices with similar energy signatures the algorithm has to differentiate, the lower its accuracy will be as the misclassifications amongst these similar devices will increase.

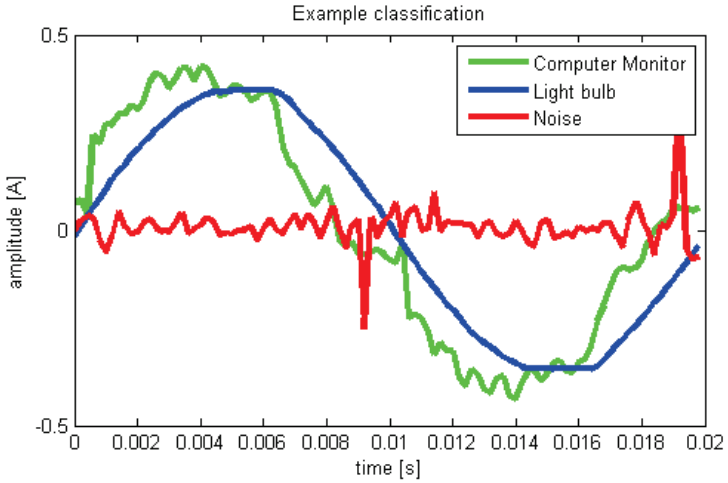

Figure 2 Sample curves of the second stage (computer monitor and noise have been trained to the algorithm, but not the light bulb)

Similar to the first stage a machine learning approach is used for device recognition in the second stage. In the following two approaches will be presented.

Approach 1: Artificial Noise Teaching

The class unknown devices is trained with artificial white Gaussian noise in several strengths. The aim of this noise signals is to train with samples that are in combination as much as possible equally distributed over all parameters in the input vector. As the feature vector contains frequency values, white Gaussian noise seems to be a good approach with its constant spectral density. So each sample has a probability higher than zero to belong to the class unknown. Additionally a device is only classified as such if a predefined confidence probability is exceeded. In the example of Figure 2 the monitor will only be recognized as monitor if the algorithm is e.g. 95% sure it really is the monitor. Hence a Bayes approach appears quite reasonable because it is based on probability density functions. Bayes networks build a probability model of each class with interferences to each other. They already perform well in the first stage. Thus, for each sample, the probability is anyway calculated for all existing classes.

Approach 2: Device set teaching

Use the entire training data set from the first stage including noise and label all data as unknown devices. Additionally measurements of the devices the user is interested in were added to the training set. In our application they were a selection of device states from the same measurements set as already used for the training of the class unknown devices. The resulting dataset is highly imbalanced: We have a large amount of data in the 'unknown device' class and only little to train the devices the user is interested in. Each sample uses a specific weight to balance the weight of each class. Exactly the same measurements were used to teach the devices of interest to the user, as they have been used to teach unknown devices as well but with different weights. Thus the algorithm deals

with contradictory information. Even the artificial noise overlaid on each measurement has been used in both classes without any change.

Class weighting

The weighting of the training data is of critical importance. In the training, each sample is weighted according to their class relevance. See formula (1) for the calculation. w_c is the weight factor of the individual sample belonging to the class c, p_c is the target weight of the class c, n_c is the number of samples in that class and n the total amount of samples in the training set.

$$w_c = \frac{p_c \cdot n}{n_c} \; for \sum_c p_c = 1 \; and \sum_c n_c = n \tag{1}$$

Weighting can only be applied to classifiers that can consider such information in their training. Different weightings have been compared in this paper. The results will show a worst case scenario for the algorithm as contradictory information has been used.

Accuracy evaluation

The second stage uses the data classified as events ($\overrightarrow{F_E}$) from the first stage. True as well as falsely classified events will therefore get processed. In formula (2) recognized events from the first stage are marked blue whereas the missed events are shown in red.

$\boldsymbol{classified\ as \rightarrow}$	$\boldsymbol{Event}$	$\boldsymbol{NoEvent}$
$\boldsymbol{Event}$	$TP\ (True\ Positive)$	$FN\ (False\ Negative)$
$\boldsymbol{NoEvent}$	$FP\ (False\ Positive)$	$TN\ (True\ Negative)$

(2)

The second stage however does not get the NL label from the first stage. So even on the first stage incorrectly classified events (FP) can be classified correctly in the 2nd stage.

Dataset

The training dataset contains 64 different device states accumulated from 32 different devices. In total the dataset contains 3177 measurements of which 1000 measurements were artificial white Gaussian noise with a variance of 80 mA. The measurements of the interested device states have been included in the training set. The amount of learning curves was on average 35 measurements. After the training process the algorithm has been tested with an independent test set of 3998 measurements containing events of the following device states:

1. Toshiba Satellite Pro C850-1dx (Laptop state charging)
2. Acer LCD Monitor T231H (Monitor state on)
3. Fust Primotecq 01/07 (Ventilator state S2)
4. Osram duluxstar 23W (Energy saver bulb state on)
5. Megaman LED Classic 8W (LED light state on)
6. Osram CLAS A CL 40W (Light bulb state on)

Results

A BFTree algorithm was trained according to the second approach with the same dataset as used in the first stage. All samples were labelled as unknown devices. This set has been extended with known devices. This approach is used in all further comparisons.

Algorithms

The recognition accuracy results are provided in Table 1 and Table 2. The columns show the chosen devices that our algorithms trained with. Comparisons with different weightings of each sample are listed in the rows. The weight of each sample is calculated according to Equation (1). In the following results, p_n is the weight of the unknown device class. It has to be mentioned that the results depend on the first stage. Thus all the missed events are not validated. The event detection reached an accuracy of 93%. See [9] for more detailed accuracy analysis.

Table 1 shows the result of the Bayes algorithm compared with different weighting functions.

Table 1 Device identification result with Bayes Net algorithm

Nr. Devices / Device weight	1	1+2	1+2+3	1+2+3+4	1+2+3+4+5	1+2+3+4+5+6
Each class equal weighted	**98.9%**	87.7%	70.9%	56.0%	41.4%	22.8%
$p_n = 30\%$	97.4%	**88.4%**	**78.4%**	**75.7%**	**60.4%**	**43.3%**
$p_n = 40\%$	98.1%	88.1%	72.4%	65.7%	55.2%	39.2%
$p_n = 50\%$	98.8%	88.1%	71.3%	61.9%	51.9%	34.0%
$p_n = 60\%$	**98.9%**	87.7%	72.0%	58.6%	45.5%	29.5%
$p_n = 70\%$	**98.9%**	87.7%	72.4%	58.6%	45.1%	26.1%
$p_n = 80\%$	98.5%	88.1%	72.4%	56.7%	43.7%	25.0%
$p_n = 90\%$	98.5%	87.7%	71.6%	54.1%	40.3%	22.0%

Table 2 shows the result of the BFTree algorithm also compared with different weighting functions.

Table 2 Device identification result with BFTree algorithm

Nr. Devices / Device weight	1	1+2	1+2+3	1+2+3+4	1+2+3+4+5	1+2+3+4+5+6
Each class equal weighted	87.3%	92.9%	88.4%	87.3%	86.9%	85.8%
$p_n = 30\%$	86.2%	90.7%	88.1%	86.2%	88.8%	85.8%
$p_n = 40\%$	86.6%	89.6%	88.4%	87.7%	86.2%	83.6%
$p_n = 50\%$	87.3%	90.7%	89.2%	87.7%	86.9%	**92.5%**
$p_n = 60\%$	87.7%	93.6%	89.2%	**94.4%**	**87.7%**	86.9%
$p_n = 70\%$	88.8%	**93.7%**	**92.2%**	86.7%	82.8%	86.9%
$p_n = 80\%$	93.7%	93.3%	88.4%	90.3%	67.9%	68.7%
$p_n = 90\%$	**96.3%**	89.6%	73.1%	73.5%	61.1%	53.3%

Class weighting

In Table 1 and Table 2 the best results for each combination of devices has been marked green. Results from the BFTree algorithm clearly suggest a relation between the weight of unknown and known devices. Equation (3) calculates the weight p_n of the class with unknown devices under the assumption that the dependency between known and unknown classes is linear. The rest is equally distributed to the number of devices. The factor f in Equation (3) is the rise of the function and must be verified. a_d is the number of trained devices.

Table 3 shows the results for $f = 7 \; and \; f = 8$.

$$p_n = \frac{f}{a_d + f} \; for \; f \geq 0 \tag{3}$$

Table 3 Device identification result with BFTree and dynamic weighting

Nr. Devices / Device weight	1	1+2	1+2+3	1+2+3+4	1+2+3+4+5	1+2+3+4+5+6
$p_n\|_{f=7}$	90.3%	93.3%	92.2%	90.3%	87.3%	92.5%
$p_n\|_{f=8}$	96.2%	93.3%	88.4%	90.3%	87.3%	87.3%

Discussion

Algorithm

Additional to recognizing trained devices, a classifier must incorporate the possibility that an event does not belong to any of the known devices. We looked at two strategies to achieve that: first, using a probabilistic and only accept the recognition result if its associated confidence value is above a predefined threshold. The second approach is to introduce a separate “unknown” device class. Both approaches were tried and the second approach was found to perform better.

When trying the probabilistic approach, we found that in our test set, the class probability value of Bayes networks had no relevance to true classification. In other words, there were as many test samples with a high confidence classified incorrectly as there were incorrectly classified samples with low confidence. Using different threshold values for the minimum confidence value did not improve the result. Because of the lower accuracy compared to the second approach, we decided to continue with the said “unknown class” strategy.

Using a BFTree algorithm in the second approach performed better compared to the Bayes Network in the first approach. The decision tree finds the most significant parameters with the Best First algorithm. The tree also weights each training instance properly even if there is contradictory information in the training set. We tested different configurations ranging from one to six device classes and reached at least a recognition accuracy of 87%.

Class weighting

The results strongly depend on the weighting of the classes. The results in Table 2 show that a constant weighting function does not achieve the best accuracy. Trained the algorithm with a dynamic weighting function according to the amount of known devices improves the accuracy considerably. In this paper, a linear weighting function is used. This function has to be empirically verified as it has to be tested with any combination of the known devices. In these results the device combination was always static to facilitate the comparison between different weighting functions. For $f = 7$ in formula (3) almost all results reached a value close or equal to the best result achieved in Table 2. Thus a dynamic weighting function guarantees a classification accuracy of 80 to 90% and therefore user acceptance [2].

Further result improvements

With further algorithm tuning the accuracy might be increased substantially. One way is to train the algorithm with different types and amounts of noise curves. This can be applied to noise itself and to any device that is trained even if it is a known or unknown device. With different noise on the raw measurement data the amount of samples can be artificially increased and therefore the weight of the single measurement decreased. This is a simple way to increase an algorithm’s accuracy. Another way is to enlarge the training set of the unknown devices and to eliminate contradictory information. Heuristics could also improve the results as they base on experience and could consider e.g. a switched on application can’t be turned on again or typically switch on duration or times could be considered.

Conclusion

The work presented in this paper describes our novel approach to better inform users about their household devices' energy consumption. We developed a NIALM algorithm which provides users with sufficiently detailed information about individual devices while guaranteeing high recognition accuracy. To recognize events where a device state changed, a two stage cascading machine learning approach is used.

In the first stage events are recognized and associated with predefined device classes. The associated information is communicated to the user. This first step can be integrated in NIALM systems by the manufacturers to help users understand their energy consumption at coarse detail.

In the second stage devices the user is specifically interested in are detected. The user has to train the NIALM system to recognize new devices of interest. This training information is added to the BFTree which achieves over 87% accuracy in our tests. We weighted the training data assuming there is a linear behaviour between the number of known devices and the weight of the unknown ones; i.e. the more connected devices the lower the weight for unknown devices. The achieved recognition rate can potentially be further increased with larger and more widespread training sets or by using heuristics. Our algorithm is able to output information as detailed as "Your Samsung Telestar MD 500 TV is switched on 4 hours a day" or "It is worth to replace your fridge with a more efficient one". In sum we showed a way to provide users with accurate and more detailed information about their household devices' energy consumption.

References

[1] VSE, "Wege in die neue Stromzukunft," Verband Schweizerischer Elektrizitätsunternehmen, 2012. [Online]. Available: http://www.strom.ch/uploads/media/VSE_Wege-Stromzukunft_Gesamtbericht_2012.pdf. [Accessed 02 02 2015].

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

[3] M. Mathis, A. Andrushevich, A. Rumsch, R. Kistler und A. Klapproth, «Improving the Recognition Performance of NIALM Algorithms through Technical Labeling,» in *Embedded and Ubiquitous Computing (EUC), 2014 12th IEEE International Conference on*, 2014.

[4] G. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE,* vol. 80, no. 12, pp. 1870-1891, 1992.

[5] J. Liang, S. Ng, G. Kendall and J. Cheng, "Load signature study x00A1;V part I: Basic concept, structure and methodology," in *Power and Energy Society General Meeting, 2010 IEEE*, 2010.

[6] J. Liang, S. Ng, G. Kendall and J. Cheng, "Load signature study x00A1;V part II: Disaggregation framework, simulation and applications," in *Power and Energy Society General Meeting, 2010 IEEE*, 2010.

[7] PlotWatt, "PlotWatt," 2013. [Online]. Available: https://plotwatt.com/. [Accessed 2 2 2015].

[8] Bidgely, "Bidgely," 2013. [Online]. Available: http://www.bidgely.com/. [Accessed 2 2 2015].

[9] M. Mathis, P. Huber und A. Klapproth, «Recognizing appliance and their events with the same NIALM algorithm,» in *8th International Conference on Energy Efficiency in Domestic Appliances and Lighting (EEDAL)*, 2015, in Press.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter,* vol. 11, no. 1, pp. 10-18, 2009.

[11] VSE, "Wege in die neue Stromzukunft," Verband Schweizerischer Elektrizitätsunternehmen, 2012. [Online]. Available: http://www.strom.ch/uploads/media/VSE_Wege-Stromzukunft_Gesamtbericht_2012.pdf. [Accessed 02 02 2015].