Frank Englert, Patrick Lieser, Alaa Alhamoud, Doreen Boehnstedt, Ralf Steinmetz:
Electricity-Metering in a Connected World: Virtual Sensors for Estimating the Electricity Consumption of IoT Appliances.. In: The 3rd International Conference on Future Internet of Things and Cloud, August 2015.

Frank Englert, Patrick Lieser, Alaa Alhamoud, Doreen Boehnstedt, Ralf Steinmetz
Multimedia Communications Lab
TU Darmstadt, Germany
Email: firstname.lastname@kom.tu-darmstadt.de

***Abstract*—Due to rising electricity prices, there is an increasing incentive to save energy. Therefore, more and more large organizations intend to reduce their energy consumption. Often, their plans cannot be realized due to missing insights into the causes energy consumption. Centralized energy meters provide no information at which appliances the energy is spent and the installation of thousands of distributed meters is often not feasible from an economic point of view. To simplify the energy metering in large scale, we propose to make Internet of Things (IoT) appliances aware of their own electricity consumption using on software based virtual energy sensors. We demonstrate how to automatically generate those energy models for nearly arbitrary networked devices with a high accuracy. Our purely software based energy metering solution approximates the energy consumption of common office equipment with an error between 2.19% and 10.8%. Using our approach, IoT appliances become aware of their own energy expenditure. This greatly simplifies energy metering on device level granularity, giving appropriate user feedback and developing more energy-efficient appliances. All these benefits are achieved without the need for installing additional hardware sensors.**

I. Introduction

As stated by Darby [1] and Fischer [2] up to 25% of the electricity consumption in home or office environments could be saved if the inhabitants of an environment would consume energy more carefully. Especially in large organizations with thousands of employees those numbers sum up to huge saving potentials.
However, the realization of those plans is rather challenging. First, there is a big stock of appliances which contribute to the total energy consumption. Second, the employees do not benefit from saving energy. Due to these facts, sophisticated tools for monitoring the consumption, analysing for saving potentials and providing saving recommendations are required for holistic energy management.
Hereby, the fine grained consumption monitoring is a very crucial part. Only if the actual consumption is recorded with an appropriate level of detail, the aforementioned steps can be applied successfully. Especially in large organizations with thousands of appliances, monitoring becomes the bottleneck for energy optimization. As we will state in section II, existing solutions either do not scale to large environments, do not provide a sufficient level of detail or are too expensive for medium to large scale use cases. Thus, we propose a software based electricity metering solution which uses virtual sensors to estimate the electricity consumption of networked appliances and Internet of Things enabled devices.
Our solution provides insights in the electricity consumption on device level granularity without requiring the installation of new or additional sensors. Instead, our purely software based electricity metering uses different observable device state variables to estimate the electricity consumption of common appliances. In this paper we focus on a method to automatically generate high accuracy energy models for a broad set of electrical appliances. More precisely, we show that our algorithm is capable of creating energy models which determines the electricity consumption of common office equipment with an accuracy of up to 97,81%. As our solution does not require an additional metering infrastructure, it scales well for large environments.
This paper is structured as follows. At first, in Section II we name and describe related work. Then we describe the design fundamentals of our virtual energy sensors. Building upon that, we describe the implementation of our generic energy model generator in Section IV and evaluate its performance in Section V. Finally, we conclude this work.

II. Related Work

Our main goal is the creation of tools which enable large organizations to gain insights in their electricity consumption. In other words, we want to provide methods for measuring the electricity consumption on appliance level with minimal effort and low installation costs. Furthermore, our proposed solution should be applicable in large scale environments with thousands of electrical appliances. To clearly argue, why our solution is required, we will analyze state of the art electricity metering approaches with regard to their provided level of detail, accuracy and installation costs.

A. Decentralized Electricity Meters

When using decentralized smart meters, one metering unit per appliance is required. This metering unit can be directly integrated in the wall outlet, it can be a special plug between the appliance and the wall outlet [3] or installed at any other location where it can sense effects of the electricity consumption [4]. Often, those metering units are interconnected via wireless communication channels. For example, the ACme [5] distributed smart meter uses 802.15.4 together with 6LowPAN for communication with other nodes and a gateway node. If not manually configured, the smart meters are not aware of the currently attached appliance. To avoid a manual

configuration, researchers have developed a machine learning based approach to automatically detect the currently attached electrical appliance [6], [7]. Those distributed smart meters provide a good accuracy together with a medium scalability for big installations.
While the hardware should also work in large scale, the broad distribution of this technology is limited by installation costs, efforts and the mobility of appliances. For bigger organizations which would need thousands of distributed smart meters, these factors are rather important. Furthermore, the insertion of sensors in the power supply of appliances might be unacceptable in some scenarios. Especially when the failure of power meters would introduce faults in otherwise smoothly running processes, those distributed smart meters might not be an acceptable solution.

B. Load Disaggregation

Centralized electricity metering requires a single electricity meter to monitor the summarized consumption of a whole circuit with all attached appliances. To split this load curve in its components, researchers have developed various non-intrusive load monitoring (NILM) algorithms. Those algorithms take the summarized load curve of all appliances in an environment and calculate a set of appliances which caused this load curve. With this technology, one can get the energy consumption of all appliances in an environment with minimal number of additional sensors. Historic approaches towards NILM tried to extract the appliances which created a load curve directly from the measured power signal [8], [9], [10]. Newer approaches reverse this principle by trying to find a matching set of electrical appliances which restore the observed load curve best. Kolter [11] proposes the usage of factorial Hidden Markov Models for this purpose and Egarter [12] experiments with the usage of genetic algorithms for selecting a matching set of appliances. As shown by Parson [13], those disaggregation algorithms can be generalized to avoid a training phase for each household. However, currently these generative approaches are limited to small scale environments with relatively few electrical appliances per circuit.
A completely different approach of load disaggregation was developed by Patel [14], [15]. His research relies on recognizing unique radio frequency noise emissions caused by different electrical appliances. Gupta [16] shows, how those signatures could be recognized by a single sensor installed in the circuit breaker box of a household. Those switch events are mapped to changes in the whole house power consumption in order to determine the power consumption on device level granularity [17]. Current research in this field focuses on scaling this technology scales with regards to the number and heterogeneity of appliances in an environment.

C. Indirect Energy Metering

The field of mobile computing makes use of indirect energy metering. All major mobile operating systems provide the functionality to view the energy consumption of currently running Apps. Having such a list, the user can see which program is emptying the battery. Building upon that, the user can decide whether or not the program should be killed to increase the remaining battery runtime. Dong [18] implemented this functionality for Linux-based machines using a principal component analysis. Zhang [19] uses a multi-variant linear regression model to estimate the power consumption of currently running Apps based on their hardware resource usage. Their power model includes the various parameters from the categories CPU, WiFi, Audio, LCD, GPS and Cellular. However, their model was manually fine-tuned for a particular hardware design but they provide an automatic way, to adapt their power model to arbitrary phones based on the battery drainage. The findings of Pathak [20] indicate that the modeling of different power states is required in order to estimate the power consumption with a high accuracy.
Building upon that, the goal of this work is to generalize those energy models. In the rest of this publication, we will show how those approaches can be extended to generate energy models for nearly arbitrary electrical appliances.

III. Concept

The goal of this work is to determine the electricity consumption for a whole environment on device level granularity. As shown in Section II existing solutions are either too inaccurate or an inherent source of complexity when deployed in large scale. Our goal is to mitigate those issues by making electrical appliances aware of their own energy expenditure. Instead of equipping each appliance wit a dedicated current sensor, we rely on software-based energy models to approximate the energy consumption based on the current mode of operation. These virtual energy sensors can be realized by sensing energy related device state variables together with an appropriate regression model for calculating the energy consumption from the sensed device state.
As more and more electrical appliances are equipped with processing and networking capabilities, a centralized service can be used to collect and process the energy expenditure for electrical appliances in a particular environment. However, the main challenges in such a scenario are most probably not located the networking domain. Thus, in this paper we focus on the information processing required build virtual sensors.

Instead of relying on dedicated measurement units for each appliance we develop a purely software based electricity metering solution. Most modern electrical appliances internally use microcontrollers which precisely track and influence the state of the appliance. This state variables correlate with the electricity consumption of the appliance and thus can be used to approximate the electricity consumption of a device. However, the set of state variables required to infer the electricity consumption is device-dependent. E.g. for a desktop computer the CPU load, the Disk IO the network utilization and other parameters are required whereas for a LCD monitor the brightness setting is sufficient. However, this working principle is of course not limited to office equipment. As every class requires a specific set of input parameters, each device class needs an own energy approximation model stating its energy consumption for a certain set of input parameters. To specifically build a model for each device, we apply different regression models from the field of machine learning. For each tuple of input parameters, such a model approximates the power draw caused by the appliance.

A. Data Collection

In our proposed solution, each appliance present in a certain environment has its own instance of an virtual energy sensor. This so called approximator estimates the energy consumption of its corresponding appliance. In order to perform this task, the approximator has to observe the state variables of its device to estimate its power draw. There exist three different possibilities to obtain these parameters. Namely, these are (1) appliances equipped with a network interface, (2) appliances equipped with a network interface and also with computation facilities, (3) and devices equipped with dedicated additional sensors to obtain their state variables. In the following Section, we will describe these possibilities.

1. Additional Sensors: There exists a big stock of legacy devices without any network interface. To integrate those appliances in our proposed solution, the idea of using opportunistic sensing must be abandoned. Rather than that, we propose the installation of cheap additional sensors to measure the energy consumption of those appliances. This very interesting approach was described by Kim [4]. The additionally installed sensors make the state variables of legacy appliance accessible from the network. For example, one can install a microphone connected to a mote next to a "dumb" refrigerator. If the refrigerators compressor is running, the microphone can record its characteristic sound. However, the installation of additional sensors will increase the complexity of the overall system and also add costs to the energy metering solution. Thus, the installation of more hardware should be avoided whenever possible. With the rise of the Internet of Things and machine to machine communication, this solution will become less important over time as more and more appliances are equipped with networking facilities.

2. Devices with network interfaces: In the first case, an electrical appliance in an environment is able to provide the required state variables via networked interface. Currently, the number of appliances with those capabilities is rare. But in the short term future, more and more appliances will be equipped with networking facilities to make their internal state accessible. An example of such a device is a microwave oven equipped with a WiFi interface to inform the user via smartphone about the currently active program and the temperature inside. Another example can be an arbitrary network printer. These types of printers allow querying all relevant information about the number and kind of print jobs over their network management interface. However, the approximator cannot run directly on those appliances, because they do not have the ability to run a custom computer program. Instead, the approximator for this device runs on another host somewhere in the local network.

3. Devices with computation facilities and network interfaces: In the second case, an electrical appliance is capable of providing the required state variables and also computation capabilities. In this case, our approximator will directly run on this device, query the state variables and determine the energy consumption of this appliance. In this case, the calculated energy consumption is forwarded to a centralized instance in this network to accumulate the energy consumption for this particular appliance. An example of such a device is a computer terminal. It has sensors to obtain its state variables, computation capabilities to run the approximator and a network interface to forward the calculated energy consumption. However, the installation of the approximator software on this machine is required in order to gather its energy consumption. In a real world deployment, an energy management system must be able to cope with frequently changing and moving appliances. This fact is no challenge at all for our proposed solution. As our energy models travels with the moving appliance, it continues to work location independent, as long as there is connectivity to the data collection unit. Even if there is no network connectivity for a certain period, the estimations could be buffered until the network becomes available again. However, depending on the requirements, it might be necessary to handover the device to another, responsible data collection unit.

B. Challenges by Hidden State

Our proposed solution faces several challenges. The most important challenge is caused by incomplete information. Neither all electrical appliances are equipped with an network interface nor all relevant state variables are exposed over those interfaces. While the falling number of legacy appliances can be equipped with distributed smart meters or *additional sensors*, the installation of those units increases the complexity of our solution significantly. On the other side, ignoring all non-networked appliances might add a significant error to the estimations carried out by our system. In addition to that, even appliances equipped with a network interface might have hidden state variables which are not exposed to the energy estimator. If those hidden variables are required for robust energy estimation, their corresponding models will yield inaccurate estimations in certain operating modes of the appliance.

IV. Virtual Energy Sensors

Our software based energy metering relies on a regression model which transfers the obtained state variables into power signals. Thus, the regression model is clearly the most important part of this work. We will briefly describe how our regression models are created, used and evaluated. While we evaluate our energy models exemplary for office equipment, the working principle is usable for arbitrary networked appliances. We kept the description of our algorithms concise as the whole source code of this project is freely available on GitHub.

A. Energy Model Generation

A flow diagram showing the method to generate an energy model for a particular appliance is shown in Figure 1. For the model generation, it is required to measure the power consumption of an appliance together with its state variables. More precisely, a set of training data for fitting multiple regression models together with a set of verification data for evaluating the performance of those regression models must be collected in order to create an accurate energy model for the device under measure. Each of those data sets is represented by a multi-dimensional time series of state variable and a one-dimensional time series with the associated power consumption. In the next step, these signals are filtered in order to smooth the recorded signals. Subsequently, the signals are aligned to remove a time lag between the power signal and the state variables. Then, different regression models for the device under measure are fitted using the training data. To

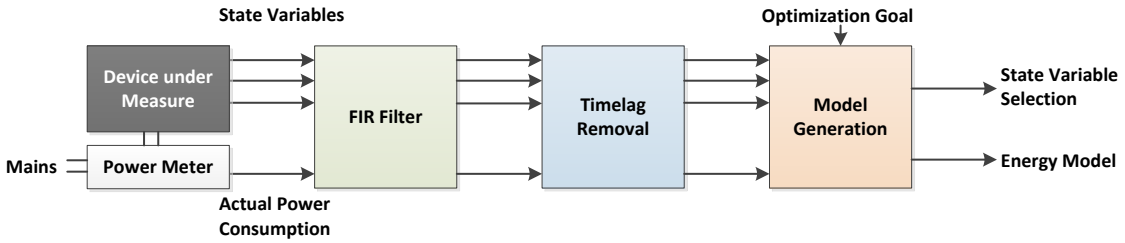

Fig. 1. Steps required for building a device specific energy model.

select the best matching regression model, the verification data together with an optimization function is used to determine the performance of each regression model. In the rest of this section, all those steps will be described with a higher level of detail.

1) Preprocessing: In order to create high accuracy energy models of nearly arbitrary appliances, a preprocessing of the measured variables is required. As shown in Figure 1, the preprocessing consists of a filtering and a time-lag removal step. The filtering step smoothen the recorded power traces as well as state variables to reduce the influence of short-term variations in the obtained signals. For this filtering task, we selected a finite impulse response (FIR) filter with a rectangular kernel function. This kind of digital filter has good numerical stability and a linear phase response. The positive influence of pre-filtering on the estimation error of our energy model for different filter sizes are shown in Figure 2. This figure clearly shows a reduction of the estimation error with increasing filter size. Furthermore, the figure shows the error introduced by the filtering itself. As shown in Figure 2, higher filter sizes reduces the estimation error of our energy model, due to the removal of high frequency components. In contrast to that, higher filter sizes distort the ground truth power signal and thus reduce the frequency resolution of our energy model. Thus, the best filter length used for pre-processing is a trade-off between good frequency resolution and a reduction of the estimation error. For our appliances under test, we selected a filter size of two. In our application scenario, this configuration provides a good trade-off between estimation error and the error introduced by the filtering.
Furthermore, the preprocessing has to correct a small time lag between the power signals obtained from the power meter and the state variable signal obtained from the device under measure. This time-lag is caused by the integrative measurement carried out in the energy meter as well as by a transmission delay between the power meter and the data collection unit. We use the cross-correlation to calculate and compensate this time-lag.

2) Model Creation: As each device class requires its own energy model, the generation of such models should happen automatically. Given a certain amount of training data together with ground truth data, state of the art regression models can be used to build such energy models. To create a particular regression model, the user has to attach a power meter to the device under observation. Now its state variables together with its actual power consumption can be collected. For best results, the user should bring the device under observation to all possible operating modes. In the next step, our system

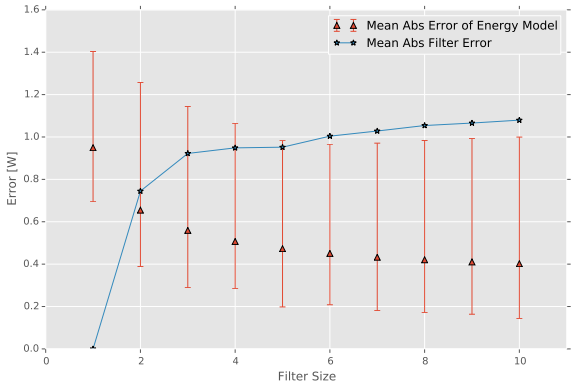

Fig. 2. Trade-off between error reduction and frequency resolution in the preprocessing stage.

automatically selects the appropriate regression algorithm and an appropriate set of features for this particular appliance and uses the collected training data to fit the model.
Currently, our solution supports eight different regression algorithms. These are listed in Table I. As each of these algorithms works best on a specific regression problem, our solution automatically finds the best possible regression algorithm for each application scenario. E.g. the best algorithm to model the energy consumption of a refrigerator might be a linear regressor whereas for determining the energy consumption of a computer terminal, a tree based regression algorithm might work better.
As the set of regression algorithms is rather limited, we selected a brute force algorithm for finding the best option. Our approach is to fit all available regression algorithms with the obtained training data. Then, in a second step, our solution calculates the performance of each algorithm using a set of verification data. The algorithm winning this competition is used to create our regression model for a certain class of devices.
For selecting a good set of features, this brute force approach would not scale. In this case the number of combinations to check grows exponentially with the number of features. To work around this issue we used a simple heuristic to select a good set of features for the regressor. This heuristic works as follows. First it calculates the correlation between each feature and the power signal obtained during training. Next, it sorts the features based on their correlation in descending order. Then, it selects the feature with the highest correlation to build a regression model and it calculates the error of this regression

TABLE I. IMPLEMENTED REGRESSION ALGORITHMS

Short Name	Regression Algorithm
ETR	Extra Tree Regressor
RFR	Random Forest Regressor
ERFR	Extra Trees Random Forest Regressor
KNNR	K-Next Neighbours Regressor
LinearR	Simple Linear Regressor
LassoR	Lasso Regressor
LarsR	Lars Regressor
DTR	Decision Tree Regressor

model. For each additional feature, the feature is only added to the selection, if it reduces the error of the existing regression model. The usefulness of this feature selection depends on the robustness selected regression algorithm and the costs of the state variable observation.
The supported set of regression algorithms covers a great variety of different usage scenarios. However, as our solution should be non-parametric, it selects the best algorithm for the current model automatically during the fitting phase of the model. To achieve this goal, our approach requires an optimization function f_o to evaluate the accuracy of the generated classifier for each regression model. To do so, we calculate a fitness score on an independent set of verification data for each regression algorithm and select the regression algorithm with the highest fitness score for our energy model.

3) Optimization Functions: We use different optimization functions to generate energy models optimized for different use-cases. In context of this work, we used four different optimization functions. Namely, these functions are: (1) Pearson Correlation, (2) Total Absolute Error, (3) Mean Absolute Error, and (4) Mean Squared Error. All of those optimization functions express different similarity measures between the real power consumption and the estimated power consumption. We will describe those functions in the following section: The *Pearson correlation* expresses the linear correlation between two variables. In our case, a high Pearson correlation indicates a high similarity between the real and the estimated power shapes. However, the Pearson correlation could not be used to indicate a constant offset between both variables.
The *Total Absolute Error* e_{tabs} is defined as the delta between the real energy consumed by the device and the approximated energy consumption. This optimization function selects the regression algorithm with a minimal energy difference to the real measurement. This optimization function will lead to energy models with minimal energy estimation errors.
The *Mean Absolute Error* e_{mabs} is defined as the average difference between the real and the estimated power consumption. It leads to energy models with minimal power estimation errors on a per sample basis.
The *Mean Squared Error* e_{mse} is defined as the squared difference between the real and the estimated power consumption. This optimization function leads to energy models with minimal power estimation errors. In contrast to the aforementioned function, the Mean Squared Error penalizes high variability disproportionately. Unless stated otherwise, this optimization function was used as default value.

4) Energy Consumption Approximation: Once a regression model is build, it can be used to approximate the energy consumption of a certain device. To do so, the actual state variables of this particular device must be observed in order to use them as input parameters for the regression model.

TABLE II. RELEVANT STATE VARIABLES FOR A LAPTOP

Parameter	Value Range	Update Frequency
CPU Load	0.0-1.0	250...1,000Hz
Disk IO	unsigned int	Each Block I/O
Net IO	unsigned int	Each Packet RX/TX
Disp. Brightness	0-1	On Change
GPU Load	0.0-1.0	1 Hz
Battery Level	0.0 - 1.0	1 Hz
Battery Charge	float	1 Hz

There is no fixed set of state variables which are usable for all device classes. Rather, the set of parameters to use, depends on the device whose energy consumption should be estimated. For example, the regression model of our computer terminal requires the state variables CPU load, Disk IO activity, network utilization, GPU utilization and the screen brightness for good approximations. On the other hand, the parameters temperature and door state are sufficient for building a feasible regression model for our aforementioned refrigerator. Of course, these parameters must be somehow observable. As installing new sensors for these parameters would violate the core idea of our paper, we require the appliances to expose those state variables via network interface.
If required state variables are not observable, they cannot be used for applying the regression. This so called hidden state variables cause a difference between the estimated power consumption and the actual power consumption of the appliance. An example of such a hidden state variable might be the thickness of the ice layer in our aforementioned refrigerator which influences its cooling efficacy.

V. EVALUATION

To evaluate our energy model generator, we conducted three different experiments. The goal of these experiments is to test the feasibility of our energy estimation approach. Those experiments are described in the following sections.

A. Evaluation Setup

Our evaluation mainly shows the feasibility of energy models for common computing hardware. We selected those devices due to two facts: First it is very easy to observe and influence their state variables. Second, their power consumption is complicated enough to make them interesting objects for our studies. However, while our evaluation is mainly focused on computing machinery, we show that our approaches are also feasible for other appliances.
In a first step, we wrote a data collection service which collects the state variables together with the energy consumption for the particular device under measure. The energy consumption of all Desktop and laptop computers was measured with a Plugwise Circle[1]. The energy consumption of the monitors was measured with a custom build high-accuracy power meter [21]. We used this power data as ground truth to fit a regression model for this particular device. We selected 1Hz as sampling rate for all observations, which is the maximal sampling rate of our Plugwise Power meters. The observed state variables in case of a computer terminal or a laptop are shown in Table II. To obtain training data, we created a small benchmark which utilizes the CPU, the network and the hard disk drive with 10 different load levels each. This benchmark has a run time of

[1] http://www.plugwise.com/

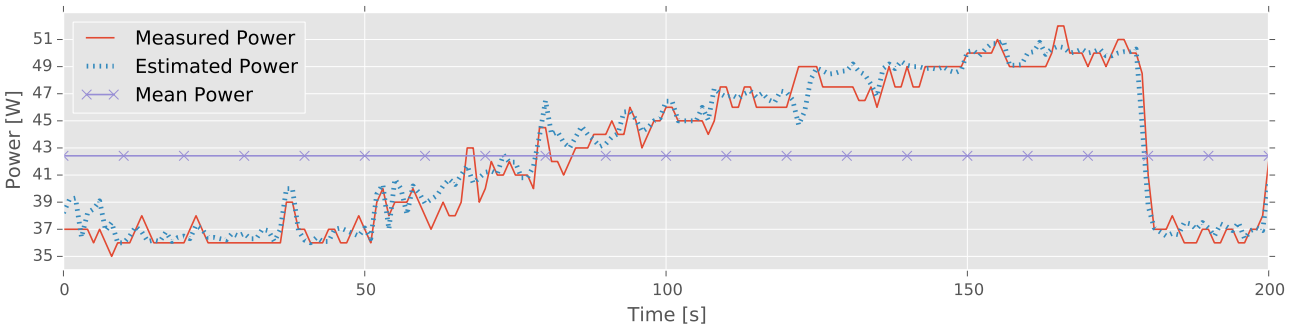

Fig. 3. The estimated power consumption using our energy follows the real power consumption with a e_{mabs} = 2.8% for a Lenovo T440s laptop with external Monitor. Estimating the power consumption using the Mean Power would cause an e_{mabs} = 25%.

460 seconds. We ran this benchmark on all machines used in the evaluation to obtain the training data sets as well as the verification data sets.

For estimating the energy consumption of the monitors, we recorded the brightness of the back light as single parameter. To determine the current brightness of a monitor, it must be connected to a computer system. Based on our decision to sample with a frequency of 1 Hz, the energy model is invoked once per second to determine the actual power consumption based on the observed state variables. To get the overall energy over a period, this time series must be integrated.

In case of our laptop or desktop computer, we have an appliance with computing capabilities. Therefore, we execute our virtual energy sensor directly on our system under test. Due to this fact, a certain error is introduced. The estimation of the actual energy consumption also causes a certain CPU load and thus certain energy consumption. Therefore, our measurement influences the quantity to measure.

In case of the office appliances, our virtual sensors were executed on a dedicated processing node. Our energy model works with all monitors supporting the DDC/CI DDC2AB protocol which is required to access the back light setting of the monitor. In theory, the regression using only the back light setting should work well for all LCD and TFT monitors. In those cases, the power consumption mainly depends on the brightness setting of the back light. However, this approach is not usable for upcoming OLED monitors. The energy consumption of those OLED displays depends on the currently shown image as each Pixel creates its own luminance. In this case either a histogram stating the color distribution of the pixels or a sub sampled version of the currently displayed image would be required as features for the regressor in order to produce accurate estimations.

B. Model Transferability

Thus, in the first experiment, we show that our generated energy models are valid for a type series rather than a single device. This property is important to keep the number of energy models within manageable bounds. In other words, if our energy models are transferable, once recorded, our energy model can be re-used for all appliances of a certain type. Thus, it is important to know, whether our energy models can be used for other devices of the same series. We call this property *Transferability*.

To conduct this experiment, we measured the state variables together with the energy consumption for 6 different but similar desktop computers. Three of them were from the same series (Lenovo ThinkCentre), two of them had an additional graphics card (also Lenovo ThinkCentre) and another computer is from a different manufacturer. During the experiment, we measured the ground truth energy consumption using Plugwise power meters together with the state variables described in table II with a sampling rate of 1 Hz. Over a measurement period of 20 minutes, we simulated various workloads by watching a film, surfing in the web, doing common office work, encoding a video and idling on the desktop.

We used the recording of one computer to fit our regression model. Hereby, our energy model generator selected the Lasso-Regression Algorithm as best selection. Then, in a second step, we validate the energy model generated before with the recordings from the other computers. The result of this validation for the Desktop computers is shown in Table III. The results clearly show robust accuracy, when the model is migrated to other machines of the same type. The model is also robust against minor hardware-changes. The error is still below 3.5% if the model is used on a computer with a changed hard disk setup. The hereby obtained accuracy is comparable with the accuracy of cheap distributed energy meters. Things look different with bigger hardware changes. The additional graphics card caused a significant change of the energy consumption which was not covered by the original model. In this case, the error is around 17%.

To determine if the energy models of computer displays are also transferable, we repeated the measurements described above for a set of four different 19” monitors of the same model series. The results are comparable with the results obtained for the desktop computer. We fitted our energy model with training data from one monitor and used the data measured from the other monitors to verify the results. Hereby, our algorithm selected a Lars Regressor. During the measurement, we changed the brightness of the back light in 10 steps from 0 to 100%. After each change of the brightness setting, we displayed different screen settings on this monitor. The first setting was a dark web site, the second setting was a rather white showing a development environment and the third image was a bluish desktop background image. As indicated by Table III, the accuracy results obtained for the monitors are comparable with the accuracy results for the desktop computer. The errors obtained in this experiment are mainly caused by hidden state. We will provide a more elaborated explanation of the reasons in section V-D.

TABLE III. RESULTS OF THE REGRESSION ALGORITHM WITH MIGRATED ENERGY MODELS FOR DIFFERENT APPLIANCES

Device	e_{mabs}	Comment
Lenovo 1	3.22%	Single HDD only
Lenovo 2	1.53%	
Lenovo 3	0.75%	
Lenovo 4	17.01%	With GPU
Lenovo 5	17.01%	With GPU
Dell 1	13.28%	Other Series
Dell A04	2.21%	Monitor
Dell A04	2.09%	Monitor
Dell A04	2.11%	Monitor
Dell A05	3.56%	Monitor

TABLE IV. RESULTS OF THE REGRESSION ALGORITHM WITH ENERGY MODELS FOR DIFFERENT NETWORKED APPLIANCES

Machine	Regressor	e_{tabs}	Class
Dell 1	ERFR	2.067 %	3
Lenovo 1	ERFR	0.683 %	3
Lenovo 2	KNNR	2.685 %	3
Macbook	LassoR	0.162 %	3
Gaming PC	LassoR	3.400 %	3
Philips Hue	KNNR	4.539%	2
Fan	RFR	1.322%	1
Canon Printer	ERFR	3.757%	1..2
Vending Machine	DTR	10.8%	1

C. Model Adaptability

There exists a uncountable number of different device series. In a perfect world, our energy model generator would be able to support all of them. However, as this goal might not be within reachability, we measure how well our model generator performs in different scenarios. We call this property *Adaptability*. If the adaptability is high, our energy model generator can be used to determine the energy consumption of arbitrary networked appliances with high accuracy. To determine the adaptability of our energy model generator, we measured the energy consumption together with state parameters for different classes (c.f. III-A) consisting of five different computer systems and four common office appliances. During the measurement, we simulated different, device specific workloads. For the computer systems, we performed the five activities surfing in the Internet, doing office work, converting video files, watching video files and idling. For each machine under measurement, we collected an independent training and test set with a length of 20 Minutes. We took care, that our testing set covers the full range of input parameters. With other words, we tried to utilize the CPU, the network interfaces as well as the available hard disks with different load parameters.

For the Philips Hue Lamp, we compared the real power consumption of the lamp with the calculated power consumption in 120 different randomly chose color and brightness settings. The fan, printer and vending machine were observed under normal office conditions for multiple hours. As those devices did not provide an interface for observing their state information, we installed an additional sensor to observe their operating mode. More precisely, we attached a microphone to those devices, recorded their noise emissions with a sampling rate of 44.100kHz. To extract features, we chopped this audio stream in small windows and extracted 13 MFCC and 10 Band Energy features from each window (c.f. [22]). Using this approach, we obtained device state information from audio emissions with a sampling rate of 10 Hz. Once this training data was available, building the energy model happened as described in section IV. However, if a device vendor decides to integrate a virtual energy sensor directly in the firmware of an electronic device, this audio based observation step is obviously not required.

The results of this experiment are shown in Table IV. A definition of this error metrics is available in section IV-A3. The results of this evaluation look promising. It is possible, to generate stable energy models for a wide set of different machines including computers, laptops, printers, lamps and vending machines using a single, non-parametric model generation algorithm. For most devices the mean absolute error e_{tabs} is below 5%.

TABLE V. LONG TERM STABILITY OF THE ENERGY MODEL

Appliance	Laptop 1	Laptop 2	Laptop 3
Samples	53,269	20,279	107,661
Manufacturer	Dell	Lenovo	Lenovo
Kind	Latitude 6440	T440s	T430s
Operating System	Linux	Win	Win
Duration [h]	14h 48m	7h 20m	29h 50m
P_{mean}[W]	29.83	37.19	15.66
e_{mabs}[%]	6.02	2.19	5.00
P_{mabs}[W]	1.8	0.81	0.78
σ [W]	3.56	0.33	0.44
Regressor	ETR	ERFR	ERFR

D. Long Term Stability

In order to test the stability under real world conditions, we conducted a long term study. We used our benchmark suit to generate training and verification datasets for three different laptops. Then, we collected the power consumption together with the state variables for a period of one month. During this time window, the laptops were used for normal office work as well as research activities during this study. The Laptops were even moved to other locations within the office on a regular basis. Having an extensive set of testing data, we used our algorithm to generate energy models for those appliances using the training and verification data set. We evaluated the accuracy of our energy model using the previously recorded test data set. To evaluate the test data, we created chunks of 60 seconds each and calculated the *Mean Absolute Error* for each chunk of estimations independently. We selected this approach to improve the visibility of time windows with high error rates. The results of this experiment are shown in Table V. The estimation errors measured in this experiment confirm the results from experiment 1 and 2. Furthermore, the standard deviation σ of the estimation error for both Lenovo laptops is rather small whereas the Dell Laptop has a much higher variance in the estimation error. This third experiment clearly shows that our software based energy estimation is usable in real world scenarios.

E. Energy Demand of Virtual Sensors

Furthermore, it is important to discuss the power consumption caused by our software based energy estimation approach. This self-consumption should be as low as possible, in order to limit the energy wastage caused by energy metering. The consumption of our approach directly depends on the number, kind and sampling frequency of observed state variables. We took no particular care to select state variables which are cheap to observe in terms of computation complexity.

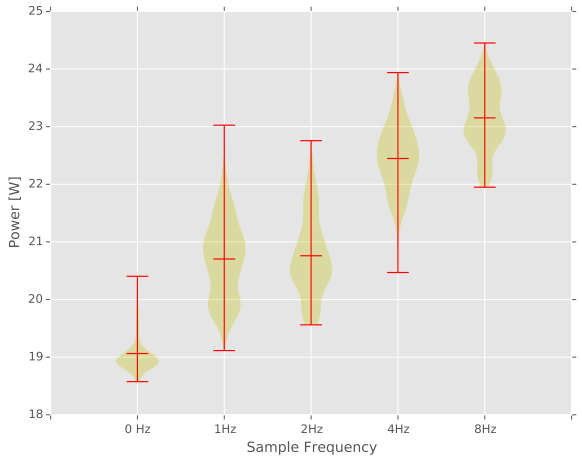

Fig. 4. Overhead caused by software based energy metering measured on Lenovo T430s under Linux.

To verify the self-consumption of software-based energy metering, we measured the difference in the power consumption of our energy estimator running at different sampling frequencies. More precisely, we executed our energy model on a Lenovo T430s under Linux. In order to get numerically stable results, we repeated this power measurement for each sampling frequency over 400 times. The results of this measurement are shown in Figure 4.
Obviously, using the energy model significantly increases the self-consumption of our system under measure. This is caused by two facts. First our implementation is currently not optimized with regard to computational efficiency. Second a broad set of state variables are observed, however most of those state variables are not used by the particularly fitted energy model. We expect a much lower overhead if only a selected set of sensors is observed and a better implementation is used.

VI. Conclusion

In this work we have presented our approach to generate energy models for arbitrary networked appliances. Instead of deploying a distributed smart meter per appliance, we measure observable state variables of this appliance to approximate its energy consumption. We have shown the feasibility of this approach exemplary for office equipment. Our implementation is capable of approximating the energy consumption of nine different appliances with an error between 2.19% and 10.8%. While currently our measurements are limited with regard to the spectrum of electrical appliances, the results indicate a broad applicability of our virtual energy sensors.
Our proposed solution can be used to determine the energy consumption of an increasing number of networked appliances. As our solution is purely software based, it could be installed or integrated in a broad spectrum of appliances with minimal cost. Thus, our solution is usable in a large scale environments for finding energy saving potentials, giving appropriate user feedback and for developing more energy-efficient appliances.

References

[1] S. Darby, "The EffectiveNess of Feedback on Energy Consumption," *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, no. April, 2006.

[2] C. Fischer, "Feedback on Household Electricity Consumption: a Tool for Saving Energy?" *Energy Efficiency*, vol. 1, no. 1, pp. 79–104, May 2008.

[3] M. Ito, R. Uda, S. Ichimura, K. Tago, T. Hoshi, Y. Matsushita, and K. Hachioji, "A Method of Appliance Detection Based on Features of Power Waveform," in *International Symposium of Applications and the Internet*, 2004.

[4] Y. Kim and T. Schmid, "ViridiScope: Design and Implementation of a Fine Grained Power Monitoring System for Homes," in *Proceedings of the 11th international conference on Ubiquitous computing*, 2009, pp. 245–254.

[5] X. Jiang, P. Dutta, and D. Culler, "Design and Implementation of a High-fidelity AC Metering Network." in *Information Processing in Sensor Networks*, 2009, pp. 253 – 264.

[6] A. Reinhardt, P. Baumann, D. Burgstahler, M. Hollick, H. Chonov, M. Werner, and R. Steinmetz, "On the Accuracy of Appliance Identification Based on Distributed Load Metering Data," 2012, pp. 1–9.

[7] T. Kato, H. Cho, and D. Lee, "Appliance Recognition from Electric Current Signals for Information-Energy Integrated Network in Home Environments," in *Ambient Assistive Health and Wellness Management in the Heart of the City*, 2009, pp. 150–157.

[8] L. K. Norford and S. B. Leeb, "Non-intrusive Electrical Load Monitoring in Commercial Buildings Based on Steady-state and Transient Load-Detection Algorithms," *Energy and Buildings*, vol. 24, no. 1, pp. 51–64, Jan. 1996.

[9] C. Laughman, K. Lee, R. Cox, S. Shaw, S. Leeb, L. Norford, and P. Armstrong, "Power Signature Analysis," *IEEE power & energy magazine*, pp. 56–63, 2003.

[10] M. Weiss, A. Helfenstein, F. Mattern, and T. Staake, "Leveraging Smart Meter Data to Recognize Home Appliances," no. March, 2012, pp. 190–197.

[11] Z. Kolter and M. Johnson, "REDD: A Public Data Set for Energy Disaggregation Research," San Diego, 2011.

[12] D. Egarter, A. Sobe, and W. Elmenreich, "Evolving Non-Intrusive Load Monitoring," pp. 182–191, 2013.

[13] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "An Unsupervised Training Method for Non-Intrusive Appliance Load Monitoring," *Artificial Intelligence*, pp. 1–42, 2014.

[14] S. N. Patel, T. Robertson, J. A. Kientz, M. S. Reynolds, and G. D. Abowd, "At the Flick of a Switch : Detecting and Classifying Unique Electrical Events on the Residential Power Line (Nominated for the Best Paper Award)," 2007, pp. 271–288.

[15] S. N. Patel, S. Gupta, and M. S. Reynolds, "The Design and Evaluation of an End-user-deployable, whole House, Contactless Power Consumption Sensor," ser. CHI '10. New York, NY, USA: ACM, 2010, pp. 2471–2480.

[16] S. Gupta, M. S. Reynolds, and S. N. Patel, "ElectriSense: Single-point Sensing using EMI for Electrical Event Detection and Classification in the Home," ser. Ubicomp '10. New York, NY, USA: ACM, 2010, pp. 139–148.

[17] J. Froehlich, E. Larson, S. Gupta, G. Cohn, M. S. Reynolds, and S. N. Patel, "Disaggregated End-Use Energy Sensing for the Smart Grid," pp. 28–39, 2011.

[18] M. Dong and L. Zhong, "Self-constructive High-rate System Energy Modeling for Battery-powered Mobile Systems," ser. MobiSys '11. New York, NY, USA: ACM, 2011, pp. 335–348.

[19] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones," ser. CODES/ISSS '10. New York, NY, USA: ACM, 2010, pp. 105–114.

[20] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang, "Fine-grained Power Modeling for Smartphones using System Call Tracing," *Proceedings of the sixth conference on Computer systems - EuroSys '11*, p. 153, 2011.

[21] F. Englert, T. Schmitt, S. Koessler, A. Reinhardt, and R. Steinmetz, "How to Auto-Configure your Smart home? High-Resolution Power Measurements to the Rescue," in *Proceedings of The fourth International Conference on Future Energy Systems (ACM e-Energy)*, no. May, 2013, pp. 215–224.

[22] F. Englert, I. Diaconita, A. Reinhardt, A. Alhamoud, R. Meister, L. Backert, and R. Steinmetz, "Reduce the Number of Sensors - Sensing Acoustic Emissions to Estimate Appliance Energy Usage," in *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings - BuildSys'13*. New York, New York, USA: ACM Press, 2013.