Alaa Alhamoud, Pei Xu, Andreas Reinhardt, Frank Englert, Philipp Scholl, Doreen Böhnstedt, Ralf Steinmetz:
Extracting Human Behavior Patterns from Appliance-level Power Consumption Data: In Abdelzaher, Tarek and Pereira, Nuno and Tovar, Eduardo: Proceedings of the 12th European Conference on Wireless Sensor Networks (EWSN 15), p. 52-67, Lecture Notes in Computer Science, v. 8965, Springer International Publishing, ISBN 978-3-319-15581-4.

Extracting Human Behavior Patterns from Appliance-level Power Consumption Data

Alaa Alhamoud[1], Pei Xu[1], Frank Englert[1], Andreas Reinhardt[2], Philipp Scholl[3⋆], Doreen Boehnstedt[1], and Ralf Steinmetz[1]

[1] Multimedia Communications Lab, Technische Universität Darmstadt, Darmstadt, Germany,
firstname.lastname@kom.tu-darmstadt.de
[2] Institute of Informatics, Technische Universtität Clausthal, Clausthal-Zellerfeld, Germany
reinhardt@ieee.org
[3] SAP AG, Walldorf, Germany
pscholl@gmail.com

Abstract. In order to provide useful energy saving recommendations, energy management systems need a deep insight in the context of energy consumption. Getting those insights is rather difficult. Either exhaustive user questionnaires or the installation of hundreds of sensors are required in order to acquire this data. Measuring the energy consumption of a household is however required in order to find and realize saving potentials. Thus, we show how to gain insights in the context of energy consumption directly from the energy consumption profile. Our proposed methods are capable of determining the user's current activity with an accuracy up to 98% as well as the user's current place in a house with an accuracy up to 97%. Furthermore, our solution is capable of detecting anomalies in the energy consumption behavior. All this is mainly achieved with the energy consumption profile.

1 Introduction

The realization of energy efficiency in buildings has become an important research topic in industrial as well as research community. The main motivation for this increasing importance is the conservation of energy in a world where energy prices are always fluctuating and very sensitive to political as well as natural crises. This is also driven by the wide spread of wireless sensor networks which made it possible to collect fine-grained data about the building context as well as the context of its inhabitants. In this paper, we develop three novel experiments which exploit the huge information provided by the smart home to achieve the main goal of our research efforts which is to conserve energy in smart homes while maintaining user comfort. The main focus of our work in this paper is the analysis of our smart home dataset which we call from now

⋆ Co-author is employed by SAP AG; however, the opinions and results expressed in this paper are his own and do not denote the point of view of SAP AG.

on SMARTENERGY.KOM dataset[4]. SMARTENERGY.KOM dataset is a large dataset which contains about 42 million data points of sensor readings and user feedback which we have collected from two smart home environments for the primary purpose of detecting human activities based on wireless sensor networks [2], thus to save unnecessary consumed energy. In the first deployment, a wireless sensor network was deployed for about 82 days. More than 22 million activity related sensor events were generated by corresponding sensors. The duration of deployment 2 was about two months, during which about 20 million sensor readings were recorded. We have used two types of wireless sensor nodes in both deployments. On one hand we deployed Plugwise[5] sensors for sensing the appliance-level power consumption of the household. Each device in the house was connected to a Plugwise sensor which measures the load of the device. On the other hand we deployed Pikkerton[6] sensors for sensing the temperature, brightness as well as the motion in the environment. In both deployments, nine daily user activities were monitored:
Deployment 1: Sleeping, Watching TV, Not at Home, Reading, Eating, Cooking, Working at PC, Making Coffee and Cleaning Dishes.
Deployment 2: Sleeping, Watching TV, Not at Home, Reading, Eating, Making Tea, Listening Radio, Slicing Bread and Ironing.
These activities have been chosen based on the available electrical appliances which can be monitored at home. Some of these activities like "Watching TV" can be directly related to the power consumption. Other activities such as "Sleeping" and "Not at Home" can be indirectly inferred from the power consumption. This list of activities does not necessarily contain all the activities performed by the user at home. Therefore, we have provided the user with the option "Ignore" which implies as a feedback that the user's current activity does not belong to the list of activities provided by us. This option helps preserving the privacy of the user as well by giving her/him the choice whether to report her/his current activity or not. All sensor readings which are related to the option "Ignore" have been excluded from the dataset before conducting our experiments. Based on these two deployments, we have built an activity detection framework which uses the feedback provided by the user to learn his current activity and relate it to the collected sensor readings. The remainder of this paper is structured as follows. Section 2 surveys related research projects whose main focus is the analysis of datasets collected by wireless sensor networks in the context of smart home. In Section 3, we present our novel concept for user localization in indoor environments based on real-time appliance-level power consumption. In Section 4, we analyze the temporal relations between the user activities and examine whether the discovered relations could increase the accuracy of our activity detection framework. In Section 5, we analyze the user's daily power consumption behavior. We conclude the paper in Section 6.

[4] The dataset is available for download under: http://www.kom.tu-darmstadt.de/research-results/software-downloads/software/smartenergykom
[5] http://www.plugwise.com/
[6] http://www.pikkerton.com/

2 Related Work

In recent years, analyzing datasets collected from wireless sensor networks in smart homes has become of great interest to computer science researchers. This is mainly driven by the great potential offered by these datasets for developing IT services which can improve the life quality as well as the energy efficiency of the smart homes. Data mining techniques have been utilized in order to extract all the possible useful hidden patterns contained in such datasets. In the work of Chen et al. [4], they analyzed a dataset which contains more than 100,000 sensor events collected from two apartments. The primary purpose of their work was to recognize human activities performed in these two apartments and understand the related energy usage. They applied clustering techniques for identifying the normal power consumption patterns, thus to detect abnormal energy usage. Using classification techniques, they trained a model for predicting the energy usage of an inhabitant based on her/his currently performed activities. Another example is given by Hoque et al. [8], where 26 days of activity related sensor events collected from a single resident home is analyzed. Based on the hypothesis that each activity will trigger a set of specific sensors, they applied pattern mining to find all simultaneously fired sensors. In the next step, different to [4], clustering is used for discovering events based on previously extracted patterns. Besides, they utilized clustering for labeling the instances. Finally, they build a classification model for recognizing the activities. Fogarty et al. [6] analyzed 3.4 million sensor readings from a home shared by two adults. Their goal was to detect water usage related activities by configuring microphone based sensors that listen for the water flow into and out of a home. They applied the classification algorithm support vector machine to train a model for recognizing different types of water usage. Fluctuations of sound waves returned by the sensors are considered as features for training the classification model. Activated sensors together with their temporal characteristics are then combined to form patterns for identifying the activities. Different from the aforementioned research projects, our analysis is conducted on a much larger dataset. Moreover, the three experiments conducted in our work have not been covered by any of these research works although similar data mining techniques are utilized.

3 Sensing Power Consumption For User Localization

User localization has always been one of the central challenges in the design of smart home environments. A wide variety of sensors such as Passive Infrared sensors can be used in order to achieve this goal. Currently, the usage of electricity consumption data for occupancy detection started to gain attention among the research community as we see in [11] where the authors used the data collected from smart meters for the purpose of occupancy detection. This leads us to the idea of utilizing new kind of sensors for user localization in smart home, namely the appliance-level power sensors which sense the power consumption of individual household appliances. Therefore, in this paper we examine the usage

of these sensors for the purpose of user localization in smart home where we aim at localizing users with a better resolution than shown in [11]. Usually, users perform specific activities in specific places, such as cooking in the kitchen, sleeping in the sleeping room and so on. Therefore, each activity is associated with certain appliances which consume energy during this activity. In other words, by knowing the devices which are consuming energy, we can infer the location of the user in the smart home. In order to verify this theory, we use supervised learning techniques where the input of the classification model will be the user's real-time appliance-level power consumption and the output is the location of the user. In the following sections, we explain the construction of the training set for the supervised learning model and we evaluate the accuracy of this model.

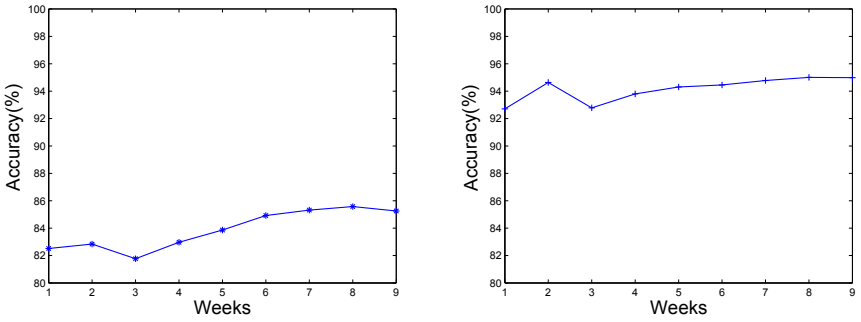
3.1 Construction of The Training Set

The first step in supervised learning is to construct a training set for building the classification model. As mentioned before, each user's location in the smart home is accompanied with a set of sensor readings representing the real-time appliance-level power consumption. These sensor readings represent the input for the supervised learning model along with the labels which represent the user's location. Sensor readings were recorded every ten seconds during the deployment. However, activities normally last for several minutes or even hours e.g. sleeping. In other words, if we directly construct a training set from these sensor readings, the size of the training set will be extremely large leading to an inefficient model construction. Therefore, we need to reduce the size of the training set without affecting the accuracy of the trained classification model. To this end, we divide the whole time series of sensor readings into timeslots of two minutes. We chose the period of two minutes as it helps achieving a good accuracy while minimizing the overlapping between activities in one timeslot. Then, for each sensor, we extracted its maximum value in each timeslot as one feature for constructing the feature vector. This means, every two minutes will represent a training instance in which the features are the maximum values of sensor readings during this timeslot. In order to provide the labels of the training instances, we relied on the user feedback which informs us about the user current activity. By knowing the current user activity, we can infer the current location of the user, because each activity is performed in one and only one location. The labeling process mainly relies on the time interval between one activity and the next activity, namely the duration of each activity. Therefore, by examining in which time interval the timestamp of an instance is falling into, we can assign the location of the corresponding activity in that time interval to the instance. The final generated form of the instances is shown in Eq. 1, where $S_{n\_max}(slot_i)$ means the maximum sensor value of sensor n in ith timeslot, and m is the total number of timeslots. Therefore, the training set is composed by a set of such instances $< I_1, I_2, ..., I_m >$.

$$Ii =< S_{1\_max}\left(slot_i\right), S_{2\_max}\left(slot_i\right), ..., S_{n\_max}\left(slot_i\right), \\ Class(slot_i) > \qquad i \in [1, m] \tag{1}$$

3.2 Building and Evaluation of The Classification Model

After obtaining the training instances, we built the prediction model for both deployments by applying the random forest classifier provided by Weka [7]. We have chosen the random forest algorithm as it proved to be the most suitable algorithm for our dataset as well as other datasets similar to it as shown in [5][16]. In order to find a good balance between accuracy and size of model e.g. to prevent overfitting the model, we first build the model with training instances of one week and then accumulate the training set by one week data points each testing. This is necessary as in real-life deployments, the learning phase should be as short as possible. Both deployments have the following four locations to be predicted, by "Outside", we refer to the instances where the user was not at home:
Deployment 1: Kitchen, Living room, Work area, Outside.
Deployment 2: Kitchen, Living room, Sleeping room, Outside.
To evaluate the built model, we apply10-folds cross validation [12] which partitions the training set into 10 subsets and always uses one subset to test the model built upon the remaining 9 subsets. This process is repeated 10 times and produces a mean accuracy over all rounds. Figure 1(a) demonstrates the accuracy of



(a) Only Plugwise sensors (b) Plugwise + time of the day

Fig. 1. Accuracy of location recognition model built for deployment 2.

the model built for deployment 2. As we can see in the figure, the random forest algorithm reaches its highest accuracy, namely 85.5% with a training set of 8 weeks. However, we can conclude from the figure that a training set of 2 weeks is already sufficient for acquiring a high accuracy. This conclusion is based on the fact that the accuracy only rises about 2.5% when the number of weeks included in the training set increases from 2 weeks to 8 weeks. This conclusion allows us to shorten the duration of the data collection process in the deployments to come which lessens the burden on the user in providing feedback and therefore leads to a more acceptance of the system. In order to obtain a better understanding of the classification accuracy, we list the precision, recall, and F-measure values

Table 1. Accuracy by classes for deployment 2 by using two weeks dataset

Classes	Precision	Recall	F-Measure
(K)itchen	87.40%	49.30%	64.10%
(S)leeping room	76.70%	99.80%	86.80%
(L)iving room	97.70%	94.10%	95.90%
(O)utside	0.00%	0.00%	0.00%

Table 2. Confusion matrix for deployment 2 by using two weeks dataset

	(K)	**(S)**	**(L)**	**(O)**
(K)	**49.33%**	44.91%	5.76%	0.00%
(S)	0.10%	**99.83%**	0.07%	0.00%
(L)	2.11%	3.82%	**94.08%**	0.00%
(O)	0.34%	99.49%	0.17%	**0.00%**

for each location in Table 1. Moreover, we show the associated confusion matrix in Table 2. The result represents the model built for deployment 2 with a training set of 2 weeks. Although the overall accuracy reached by this model is 83% (cf. Figure 1(a)), the recall values of the classes "Kitchen" and "Outside" are very low with 49.3% and 0% respectively as shown in Table 1. In order to understand the reasons for this phenomenon, we have to look on the confusion matrix in Table 2. From the confusion matrix, we can see that 44.91% of the instances of the class "Kitchen" have been falsely classified as "Sleeping room" instances. Besides, almost all the instances of the class "Outside" have also been falsely classified as "Sleeping room" instances. This can be explained based on the following facts. First of all, the confusion between the classes "Outside" and "Sleeping room" can be returned to the fact that when the user is outside or sleeping, all Plugwise sensors were almost keeping in silence as no appliances are required to perform these activities. Although, there are some values of Plugwise sensors (e.g. lamp sensor) related to the "Sleeping room" class stored in the dataset, the lamp was in most cases not turned on while sleeping. Furthermore, the instance of the class "Outside" were classified as "Sleeping room" and not the other way around because "Sleeping room" is a dominant class. This is due to the fact that the duration of sleeping is much longer than that of being outside in this deployment which leads to more training instances for the class "Sleeping room" than for the class "Outside". The activity of "Eating" was the major reason of falsely classifying instances of "Kitchen" into "Sleeping room". This activity is supposed to be identified through the Plugwise sensor connected to the radio in the kitchen. However, the radio was not always turned on or only turned on for a part of time during the activity of "Eating". To solve this problem, we need a strong discriminator which can help distinguishing the classes "Outside" and "Sleeping room". We thought about a feature which can be used in the learning process in order to achieve this task. One feature which can fully perform this role is the time of the day. By using the time of the day as a feature for building the machine learning model, we add a strong discriminator especially between

Table 3. Accuracy by classes for deployment 2 by using two weeks dataset (with time)

Classes	Precision	Recall	F-Measure
(K)itchen	81.80%	73.30%	73.30%
(S)leeping room	98.20%	99%	98.60%
(L)iving room	96.10%	96.80%	96.50%
(O)utside	83.50%	86.30%	84.90%

Table 4. Confusion matrix for deployment 2 by using two weeks dataset (with time)

	(K)	**(S)**	**(L)**	**(O)**
(K)	**73.32%**	9.59%	4.99%	12.09%
(S)	0.99%	**98.97%**	0.03%	0.0%
(L)	0.65%	0.0%	**96.84%**	2.50%
(O)	7.77%	0.33%	5.57%	**86.31%**

the classes "Outside" and "Sleeping room". We use the "hh:mm:ss" time format as Weka can deal with this time format automatically. After using the time as a feature in addition to the previous features, the overall accuracy of the model has increased as shown in Figure 1(b). To better understand the effect of adding the time to the feature set, we present the precision, recall, and F-measure values for each location in Table 3. Furthermore, we present the associated confusion matrix in Table 4. The results in these two tables have been achieved for deployment 2 with a training set of 2 weeks. As we can see from Table 3 and Table 4, the time has functioned as a strong discriminator between the class "Sleeping room" and the classes "Outside" and "Kitchen" respectively. The use of the time as a feature makes it easy to solve the confusion between the class "Sleeping room" and the class "Outside" as the user in deployment 2 always goes outside during the day and not during the night. The location "Kitchen" can also benefit from the usage of time as a feature, because the user performs most of his activities in the kitchen during the day.

4 Mining Human Behavioral Patterns

As humans tend to follow a regular routine in their daily life, their everyday activities tend to happen in a certain order which mostly repeats itself everyday. Discovering temporal relations between these daily activities may assist in enhancing the accuracy of our activity detection framework. Hence, in this section, we first try to detect any behavioral patterns which might exist in the data collected in both deployments and then we examine whether these detected patterns can help increasing the accuracy of the activity detection framework we have previously developed.

4.1 Extraction of Temporal Activity Patterns

As mentioned in [3], temporal relations between two activities (A, B) can be represented as A happens after B, before B, overlaps with B and so on. According to the user feedback in both deployments, activities were performed consecutively one after another which was a precondition for our dataset. Hence, we only examine the "before" and "after" relations between two activities. In the following section, we introduce four terms related to the analysis before explaining the operations of the pattern mining process.

Episode: According to [15], an episode is characterized by a pair of begin and end timestamps, during which one or more activities can happen. As the user's daily activities are the major interest of our analysis, we specify the duration of an episode as a single day. Hence, an episode is composed of all activities performed during the day, namely all the activities between timestamps [00:00:00, 23:59:59]. This concept is expressed in Eq. 2, where A represents one activity, T is the associated timestamp, n is the number of activities of the day, while d refers to the number of days in that deployment. After that, we construct an episode dataset by collecting all episodes during the whole deployment. By examining the dataset, we obtained 64 valid days for deployment 1 and 61 valid days for deployment 2. Days of deployment 1 are much less than the actual duration of the deployment (about 82 days). This is due to the fact that the feedback was not provided by the user in the last 18 days.

$$Episode_i =< A_1(T_1), A_2(T_2), ..., A_n(T_n) > \qquad i \in [1, d] \tag{2}$$

Sequence: A sequence is formed by at least two successively performed activities. For instance, <Eating, WatchingTV> means the activity "Watching TV" happens directly after "Eating".

4.2 Apriori Algorithm

For the extraction of the temporal relations between activities, we apply the Apriori algorithm [1] which aims to discover frequent activity sequences based on what is called their support and confidence:

Support: In our case, support measures the frequency of an activity sequence appearing in the episodes dataset. It is computed as the number of episodes that contain this sequence, divided by the total number of episodes (Eq. 3). A frequent sequence can be defined as a sequence whose support is larger than a predefined threshold (minSupp).

$$Support(< A, B >) = \frac{\#episodesContaining < A, B >}{\#episodes} \tag{3}$$

Confidence: It represents the dependency between two activities i.e. the probability that one activity occurs given that a certain previous activity has occurred. Hence, confidence for the sequence <A, B> is computed as the support of <A, B> divided by the support of A. The main principle of the Apriori algorithm is

Table 5. Examples of temporal activity patterns

A	Cooking	Eating	Making Coffee
B	Eating	WorkingAtPC	Eating
Supp. (%)	25.0	51.6	28.2
Conf. (%)	84.2	67.4	53.1

to scan the whole episodes dataset in order to find all frequent items (activities) and exclude those which are rarely performed. However, a rarely performed activity does not necessarily imply the nonexistence of a regular temporal activity pattern which involves this activity. An example from our dataset is the "Reading" activity. This activity has a support of 4.7% in deployment 1. However, if the user always sleeps after reading, then the sequence <Reading, Sleeping> can also be considered as a meaningful pattern due to its high confidence. As shown above, a threshold has to be specified which determines the minimum value the support of a sequence should have in order to be considered by the Apriori algorithm as a regular sequence. This threshold is called the minSupp. On one hand, a high minSupp value might cause the exclusion of meaningful patterns because it involves activities with low support value. On the other hand, a small minSupp value might cause the generation of a numerous number of meaningless patterns by the Apriori algorithm. In order to overcome this problem, we utilize the multiple minimum supports mechanism [14]. This mechanism assigns a $miniSupp_i$ to each item (A_i) by multiplying a user defined global miniSupp by the item's own support as shown in Eq. 4. By doing this, useful patterns regarding to the rarely performed activities will not be neglected during the process. Meanwhile, patterns regarding to one activity with support lower than the assigned minSupp will be filtered out. Therefore, we define the global_minsupp as 18%.

$$minSupp_i = global\_minsupp \times support(A_i) \tag{4}$$

By applying Apriori algorithm, we obtained a list of temporal activity patterns for each deployment. Table 5 lists some of the extracted patterns from the first deployment where A denotes the previous activity and B denotes the current activity. As we can see from the table, the user usually starts with the eating activity directly after cooking with a confidence of 84.2%. After eating he often works at PC with a confidence of 67.4%. The activity after making coffee is also eating with a confidence of 53.1%. These examples show the existence of a certain routine in our daily life. In the following section, we use the extracted patterns in the activity detection process in order to see if it can help improving the accuracy of this process.

4.3 Utilizing Patterns in Activity Detection

In this step, we integrate the patterns extracted by Apriori algorithm as extra features in building the activity prediction model. The features we used for the activity detection process were the maximum sensor values (Plugwise and

Table 6. Activity detection accuracy with and without patterns (random forest)

	Deployment 1		Deployment 2	
	Accuracy	F-Measure	Accuracy	F-Measure
without	92.8%	92.7%	97.3%	97.3%
with	96.1%	96.1%	98.3%	98.3%

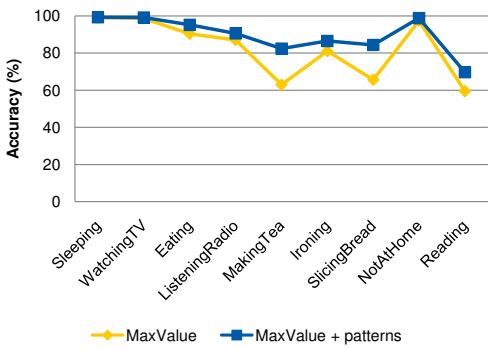


Fig. 2. Activity detection accuracy by classes, results of with and without patterns are compared.

Pikkerton) in timeslots of two minutes. As extra features, we added the previous activity combined with the most likely current activity as it appears in the sequences extracted by Apriori algorithm. Hence, the new feature vector is a combination of all these features as shown in Eq. 5, where A_{n-1} and A_n represent the previous and the most probable current activity respectively. For labeling the instances, we use the user feedback denoted as $Class(slot_i)$.

$$\begin{aligned} Instance_i =< & S_{1\_max}(slot_i), S_{2\_max}(slot_i), ..., S_{n\_max}(slot_i), \\ & A_{n-1}, A_n, Class(slot_i) > \qquad i \in [1, m] \end{aligned} \tag{5}$$

Table 6 shows the accuracy of the activity detection process for both deployments before and after adding the patterns extracted by Apriori algorithm. The used classification algorithm is random forest. The overall accuracy explicitly increases after adding the patterns. Furthermore and in order to obtain a more comprehensive representation of the classification accuracy for each individual activity, Figure 2 shows the results coming from deployment 2 using random forest. As we can see in Figure 2, the detection accuracy of each activity has also explicitly increased after adding the patterns. The reason for this improvement is that, besides the intrinsic features of the activities, namely the sensor readings, the machine learner will also learn the temporal relations between the activities from the patterns, thus recognize activities that occur in certain patterns more accurately.

5 Analysis of The Daily Power Consumption Traces

In this section we focus on the analysis of daily power consumption traces in our two deployments. The main goal is to understand the daily power consumption of individuals in smart homes and to find out based on time series analysis if the people follow a regular power consumption pattern which repeats itself over the days. The result of this analysis can be of great importance in many application scenarios: it can help developing applications which allow individuals living in smart homes to inspect their energy usage over the time leading to a more energy-aware power consumption behavior. Besides, it can be of great benefit to utility companies which by knowing the power consumption behavior of their customers can recommend a more suitable tariff and direct the smart grid to work more efficiently, thus to save energy.

5.1 Obtaining Hourly Power Consumption of Each Day

For the analysis to be conducted, we first need to compute the hourly power consumption of each day. We calculate the power consumption for each hour by summing up the power consumption within all timeslots of two minutes in that hour. However, since Plugwise values are stored in unit "Watt", we need to convert them into "Wh" (Watt hour) for acquiring the power consumption. To do this, we first compute the associated power consumption of each Plugwise sensor in each timeslot. This is achieved by averaging the readings of each sensor in that timeslot and dividing the average value by 30. We divided the average by 30 as we only need the power consumption in timeslots of two minutes. Then, for obtaining the total power consumption in each timeslot we sum all the converted values of the Plugwise sensors in that timeslot. The computation is indicated in Eq. 6, where j denotes the sensorId, i denotes the ith timeslot, $S_j(Slot_i)$ denotes the average value of sensor j during timeslot i, and n denotes the number of sensors. To compute the total power consumption in an hour, we sum up the power consumption in all timeslots within that hour as indicated in Eq. 7 where m denotes the number of timeslots, and h denotes the hour of the day.

$$P_{slot_i} = \frac{\sum_{j=1}^{n} S_j(Slot_i)}{30} \tag{6}$$

$$P_h = \sum_{i=1}^{m} P_{slot_i} \qquad h \in [1, 24] \tag{7}$$

To make the results more comprehensive, we plot the obtained hourly power consumption for each day. Figure 3 shows an example in which we see that the user consumes more energy from 01:00 pm to midnight than from 3:00 am to noon. When comparing this distribution to another one obtained from the same deployment as shown in Figure 4, we can easily observe the similarity between these two distributions. In both days, the user followed a similar power

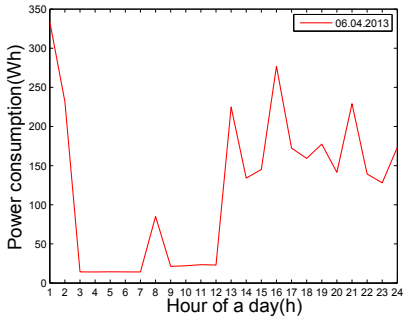


Fig. 3. Power consumption with regard to the hours of the day on 2013-04-06 from deployment 1.

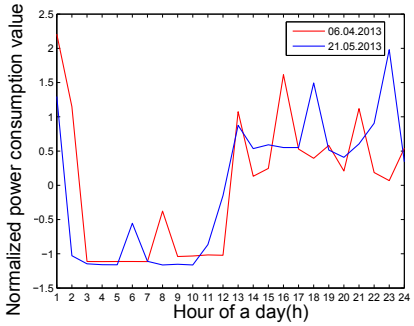


Fig. 4. Hourly power consumption distributions on 2013-04-06 and 2013-05-21 from deployment 1, consumption values are normalized.

consumption pattern only shifted in time. The values in the figure are all normalized so that they have a mean of zero and a standard deviation of one for the purpose of comparison. Based on our observation from Figure 4, we conduct a similarity comparison process on all the distributions which belong to the same deployment. By verifying that all the distributions from the same deployment follow some level of similarity, we can prove the user to have a regular power consumption behavior which is the goal of this experiment as stated before. In the following section, we introduce the process of similarity comparison, the used algorithm, as well as the obtained results.

$$Similarity = \frac{1}{1 + warping\_score} \tag{8}$$

5.2 Similarity Comparison

Similarity comparison between two time series can be conducted using several algorithms. One of the these popular algorithms is the approach of symbolic

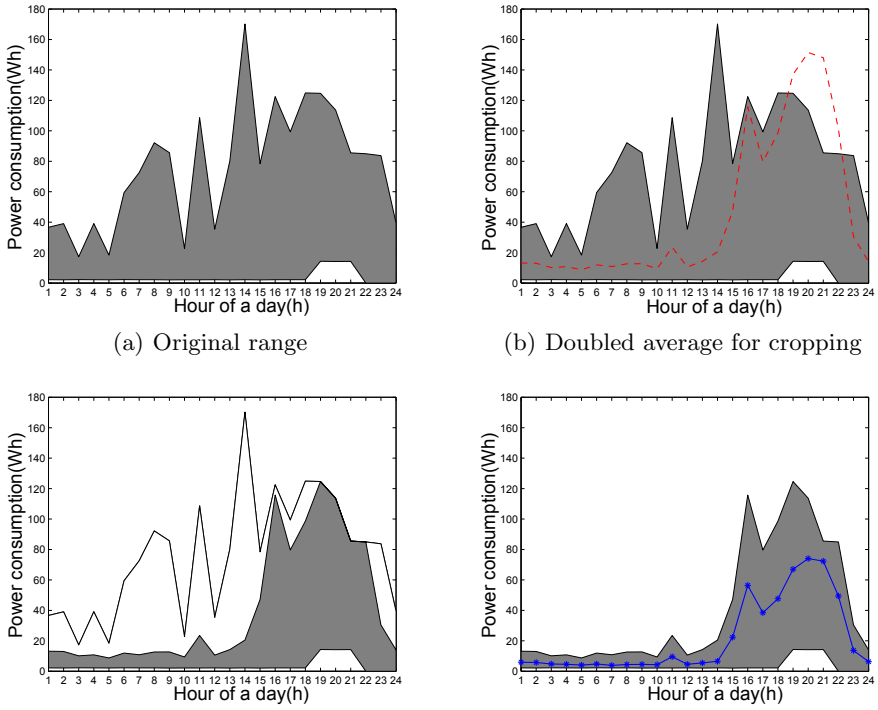
Table 7. Results of similarity comparison of power consumption distributions in each deployment

	Deployment 1	**Deployment 2**
Min_similarity	81.72%	88.13%
Max_similarity	97.13%	97.68%
Avg_similarity	90.32%	93.50%

representation [13] in which we convert each time series into a sequence of symbols and calculate the distance of these resulting sequences of symbols. The main disadvantage of this approach is that it does not take time shifting into consideration. Thus, it will not recognize two series like the ones shown in Figure 4 similar to each other only because they are shifted in time. In order to address this problem, we apply the Dynamic Time Warping (DTW) algorithm [10] which aims to find the best alignment between two time series. The result is represented by a warping path that indicates how each point of one distribution is aligned to the point of another distribution. Besides, it also produces a warping score to indicate the distance between two distributions after the alignment. In order to verify the similarity between each pair of distributions, we converted the warping score into a similarity measure based on Eq. 8 as clarified in [9]. The result after applying the DTW algorithm is a set of similarity values coming from the warping scores after comparing all the distributions. Table 7 summarizes the minimum, maximum as well as the average similarity obtained from both deployments. As shown in the table, the power consumption distributions in deployment 1 are at least 81.72% similar to each other while the minimum value in deployment 2 reaches a similarity value of 88.13%. The maximum values in both deployments exceed 97%. Moreover, daily power consumptions in deployment 1 are 90.32% similar to each other on average. The average value reaches 93.50% in deployment 2. As a result of this analysis we can conclude that the daily power consumption in both deployment follows a regular pattern which confirms the fact that the inhabitants in both deployments tend to consume power in a regular pattern which repeats itself everyday. Additionally, as similarity values from deployment 2 are higher than those from deployment 1, we can say that the user in deployment 2 tends to have a more regular power consumption behavior. In order to verify these results, we conducted a further analysis in the following section in which we examine abnormal power consumption values which occur very rarely in both deployments but might contradict with our conclusion in this section. By examining these values and showing that they are rare and untypical, we make our conclusion in this section more reliable.

5.3 Further Analysis

In order to filter out abnormal power consumption behavior, we extracted the minimum and maximum power consumption of each hour over the whole deployment. Using these values we formed an area as shown in Figure 5(a) where the



(a) Original range (b) Doubled average for cropping

(c) Cropped range (d) Normal power consumption pattern

Fig. 5. The generation of normal power consumption range for deployment 2.

x axis represents the hour of the day, and the y axis represents the associated power consumption. By doing this, all power consumption values are ensured to be contained in this area. Figure 5(a) is generated from the values of deployment 2. As we can see from the figure, although the daily energy distributions were verified to follow similar trend, they fluctuate in a certain range. At some point in time, the fluctuation is especially large. For instance, value at 14:00 varies from 0 to 160Wh. In order to verify whether the peaks in Figure 5(a) are only outliers and do not represent the regular power consumption behavior, we defined an empirical threshold which is equal to the double of the average hourly power consumption. If the power consumption at a certain hour exceeds this threshold, this consumption is considered to be an outlier and thus should be excluded from the dataset. The threshold for cropping the area is indicated in the dashed line in Figure 5(b). Figure 5(c) is the cropped area which covers the majority of the power consumption values. As we can see from the figure, a part of the area was cropped out, especially the peaks. This verifies that the peaks are actually the abnormal power consumption values and do not reflect the regular consumption pattern. Figure 5(d) depicts the remaining area after cropping. The dotted line with the asterisks indicates the average value after

removing the outliers. As shown in the figure, the power consumption keeps low from the beginning of the day to the midday. There are several reasons for this phenomenon. First of all, the user normally sleeps during some hours of this range which leads to almost no power consumption. Furthermore and according to the user feedback, the user in deployment 2 used to get up early. The activity after getting up was either eating or going out with both activities having low power consumption. Although the radio was used sometimes during eating, only small amount of energy is required for this activity. Another activity which happens in the morning is "Making Tea". Although this activity consumes a high amount of energy, it was only performed three times during the whole deployment with a short duration. This also explains why some outliers existing during this period. The higher power consumption in later hours is mainly due to the activity of "Watching TV". The extracted power consumption range can be of great benefit and importance in many application scenarios. One application scenario is security combined with energy conservation in which the user can be alerted if her/his real-time power consumption exceeds the normal power consumption area.

6 Conclusions

In this work we presented three experiments conducted on our SMARTENERGY.KOM dataset. In the first task, we successfully built a classification model that is able to predict a user's current location based on his real-time power consumption. In the second experiment, we extracted the temporal relations between the activities performed in each deployment. Furthermore, we showed that these temporal patterns can be treated as features for improving the accuracy of our activity detection platform. In the third experiment, we studied the distributions of daily power consumption with regard to the hours of the day. By comparing the similarity of these distributions we showed that the user in each deployment has a regular power consumption behavior. Moreover, we extracted the normal power consumption pattern for each user which can be of great benefit in many application scenarios. Our solution is capable of determining the user's location, activity as well as common patterns. All this information is mainly mined from electricity consumption of common home appliances. Thus, our work is a strong foundation for energy consumption feedback systems and represents the next important step towards energy management systems without a human in the loop.

7 Acknowledgments

This work has been financially supported by the German Research Foundation (DFG) in the framework of the Excellence Initiative, Darmstadt Graduate School of Excellence Energy Science and Engineering (GSC 1070) and has been cofunded by the Social Link Project within the Loewe Program of Excellence in Research, Hessen, Germany.

References

1. R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. 1994.
2. A. Alhamoud, F. Ruettiger, A. Reinhardt, F. Englert, D. Burgstahler, D. Bohnstedt, C. Gottron, and R. Steinmetz. Smartenergy.kom: An intelligent system for energy saving in smart home. In *Local Computer Networks Workshops (LCN Workshops), 2014 IEEE 39th Conference on*, pages 685–692, Sept 2014.
3. J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579, 1994.
4. C. Chen, D. J. Cook, and A. S. Crandall. The user side of sustainability: Modeling behavior and energy usage in the home. *Pervasive and Mobile Computing*, 9(1):161–175, 2013.
5. F. Englert, T. Schmitt, S. Koessler, A. Reinhardt, and R. Steinmetz. How to autoconfigure your smart home?: High-resolution power measurements to the rescue. In *Proceedings of the Fourth International Conference on Future Energy Systems*, e-Energy '13, pages 215–224, New York, NY, USA, 2013. ACM.
6. J. Fogarty, C. Au, and S. E. Hudson. Sensing from the basement: a feasibility study of unobtrusive and low-cost home activity recognition. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 91–100. ACM, 2006.
7. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explor.Newsl.*, 11(1):10–18, Nov. 2009.
8. E. Hoque and J. Stankovic. AALO: Activity recognition in smart homes using active learning in the presence of overlapped activities. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 6th International Conference on*, pages 139–146. IEEE, 2012.
9. Z. C. Johanyák and S. Kovács. Distance based similarity measures of fuzzy sets. *Proceedings of SAMI*, 2005.
10. E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005.
11. W. Kleiminger, C. Beckel, T. Staake, and S. Santini. Occupancy detection from electricity consumption data. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, pages 10:1–10:8, New York, NY, USA, 2013. ACM.
12. R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, volume 14, pages 1137–1145, 1995.
13. J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11. ACM, 2003.
14. B. Liu, Y. Ma, and C.-K. Wong. Classification using association rules: weaknesses and enhancements. In *Data mining for scientific and engineering applications*, pages 591–605. Springer, 2001.
15. D. Lymberopoulos, A. Bamis, and A. Savvides. Extracting spatiotemporal human activity patterns in assisted living using a home sensor network. *Universal Access in the Information Society*, 10(2):125–138, 2011.
16. 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. In *Sustainable Internet and ICT for Sustainability, 2012*, pages 1–9, Oct 2012.