Energy Efficiency (2020) 13:1633–1651
https://doi.org/10.1007/s12053-020-09884-2

SHORT COMMUNICATION

A deep learning approach for anomaly detection and prediction in power consumption data

C. Chahla · H. Snoussi · L. Merghem · M. Esseghir

Received: 14 March 2019 / Accepted: 13 July 2020 / Published online: 7 August 2020

Abstract Anomaly detection in power consumption data can be very useful to building managers. It allows them to detect unexpected power consumption values, identify unusual behaviors, and foresee uncommon events. This paper proposes a novel unsupervised approach to detect anomalies in power consumption data. We combine the clustering-based methods with the prediction-based ones to learn typical behavior scenarios and to predict the power consumption of the next hour. These scenarios are explored by applying the K-means algorithm on 24 different K-means groups representing the 24 h of the day. This is based on the assumption that identical daily consumption behavior can appear repeatedly due to users' living habits. In order to detect the anomaly 1 h before its occurrence, a Long Short-Term Memory (LSTM) has been trained to predict the next power consumption value. This predicted value with some earlier data values are concatenated into a vector then compared with the learned typical scenarios. We used Auto-Encoders to detect anomalous days in general and this novel method to specify at what time the anomaly has occurred. Our approach not only detectss anomalies in off-line mode but also allows real-time detection on live data streams.

Keywords Anomaly detection · K-means · LSTM · Auto-Encoders · Power consumption

Introduction

Anomalies in data are patterns that do not match the well-defined notion of normal behavior (Chandola et al. 2009). Anomaly detection in power consumption data can be very effective, making energy-efficient home improvements as well as saving cost. Residential and commercial buildings consume 60% of the world's total amount of electricity (UNEP 2017). Lighting, heating, and cooling constitute the biggest part of energy consumption. But lighting all alone is the most important energy usage (Energy 2010). Techniques utilized for reducing the energy waste nowadays, for example, rely on motion detector for each source of light switching them on and off.

In general, the electric power demand is highly affected by weather conditions. In summer, the growth of power demand on the consumer side can be correlated with cooling demands. Similarly, in winter, power consumption increases because of heating needs. The power consumption varies also during

C. Chahla (✉) · H. Snoussi
University of Technology of Troyes, Institute Charles Delaunay-LM2S, 12 Rue Marie Curie, Troyes, France
e-mail: charbel.c@hotmail.com

L. Merghem · M. Esseghir
University of Technology of Troyes, Institute Charles Delaunay-ERA, Troyes, France

the same weather between weekends and weekdays. Thus, anomaly patterns would differ from weekdays to weekends.

In fact, anomalies can be classified into three categories (Chandola et al. 2009): (1) Point anomalies: when an individual point is anomalous with respect to the rest of the points. For example, a daily cooling energy consumption can be very high comparing with previously recorded values. (2) Contextual anomalies: when the data is considered anomalous in a context but normal in another. For example, a heating consumption value might be normal in winter but not in summer. (3) Collective anomalies: if a group of data instances is not normal in comparison with the entire dataset, it is considered as collective anomaly. While point anomalies can happen in any type of dataset, collective anomalies can only happen in datasets where the data are related.

Finding abnormal behaviors helps to reduce wasting energy and the cost of energy for the consumers. Anomaly detection is considered a very important task for improving energy efficiency by identifying unusual behavior, e.g., forgetting to turn off stoves after cooking. Figure 1 shows an example of unusual energy consumption for a user in 3 consecutive days. Another major issue that can be identified through anomaly detection techniques is power theft. In developing countries, 50% of the generated energy is lost by power theft as reported by the World Bank (Antmann 2009).

Anomalous feedbacks can also be used to alert consumers and to give them assistance towards identifying faulty equipment or misconfigured machines consuming more energy than required. Moreover, anomaly detection in power consumption data can be used to supply precise demand-response plans to customers (Zhang et al. 2011). Finding anomalies in energy consumption relies on identifying patterns in data and many data mining and statistical approaches have been applied to identify these patterns, e.g., Zhang et al. (2011), Nadai and van Someren (2015), Liu et al. (2011), and Chou and Telaga (2014). The majority of these approaches propose to detect anomalies in offline mode since they have to handle a large amount of data (Lee et al. 2001).

Auto-Regressive (AR), ARIMA (Auto-Regressive Integrated Moving Average), and SARIMA (Seasonal ARIMA) are some well-known time series forecasting techniques that explores time series based on their own past values. They were initially proposed by (Box et al. 2015) and they were successfully used in forecasting economic, marketing, social problems, etc. The basic idea of these methods relies on using their own lags and the lagged forecast errors to explore future behaviors. The Auto-Regressive technique considers that the present value of the time series at time t is a ponderation between all previous values of this series and some weight factors. Similarly, the Moving Average (MA) tries to model the series as the weighted sum of random shocks or error terms. ARIMA model combines of the MA model and the AR model, then it integrates into it the transformations needed to convert the series into a stationary one. Finally, if a time series

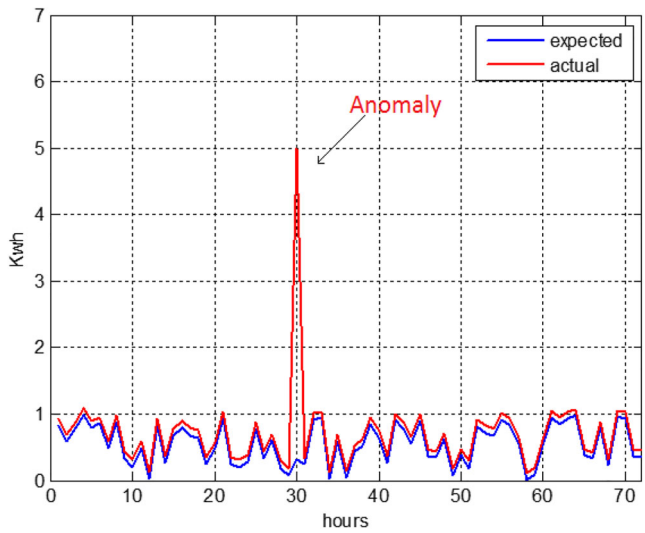

Fig. 1 An example of energy consumption in 3 consecutive days

exhibits potential seasonality, SARIMA can be used. Instead of subtracting consecutive terms, SARIMA uses seasonal differencing.

Contribution

In this paper, we proposed a novel approach for anomaly detection in power consumption data, making the following contributions:

- We proposed a new approach combining the LSTM (Long Short-Term Memory) algorithm with the K-means algorithm in order to detect and to predict anomalies in power consumption data. The motivation behind combining clustering methods with prediction-based ones relies on (1) the assumption that the power consumption of a user switches between different typical behaviors. These behavior scenarios can be learned through the K-means algorithm. (2) On the other hand, the behavior slightly changes over time and recent data weights more than old data. Thus, the power consumption relative to distant periods is not crucial for prediction and the predicted values should depend on the most recent seen values.
- We compared the performance of the proposed approach with the individual LSTM and K-means clustering applied separately, showing the superiority of the proposed method. In fact, the predicted values using the proposed approach are more close to the real power consumption data than that of the single K-means or the single LSTM. This allows less false alarms.
- An Auto-Encoder was used in order to visualize the data and to detect anomalous days in general without specifying at what time the anomaly has happened. This can be useful to building managers to analyze users' behaviors in the past.
- In order to compare with other state of the art methods, we manually inserted outliers and we considered them true anomalies. The experiments show the superiority of our method in terms of accuracy and its ability to reconstruct the data.

This paper is organized as follows: the "Background information" section provides a brief description of the Auto-Encoder, the K-means, and the recurrent neural network. The "Proposed method" section presents our proposed method. In the "Data set and behavior analysis" section, we describe the dataset used. The "Results and discussions" section shows the results with some discussions. Finally, the "Conclusion and future work" section provides some concluding remarks.

Background information

In this section, we provide the reader with a general overview on the machine learning approaches used in this study: Auto-Encoder, LSTM, and K-means. Auto-Encoders are used in our study to detect anomalous days whereas the LSTM and the K-means can be used to detect anomalies in real time and to timely localize the anomaly.

Auto-Encoder

The architecture of a basic Auto-Encoder (Bengio 2009) is presented in Fig. 2. Layer L_1 is the input layer, L_2 is the hidden layer, and L_3 is the output layer. The Auto-Encoder is an unsupervised artificial neural network that is trained to learn an approximation to the identity function. In other words, it tries to reproduce the output vector $\hat{x} = [\hat{x_1}; \hat{x_2}, ..., \hat{x_N}]$ that is similar to the input vector $x = [x_1, x_2, ..., x_N]$. In the training phase, the Auto-Encoder is forced to learn a "compressed" representation of the input through the hidden layer to reconstruct the the output. Actually, this architecture often ends up learning a lower dimensional subspace very close to PCA (principal component analysis). In order to detect anomalies, Auto-Encoders are normally used where the objective is to rebuild the inputs by minimizing the reconstruction error in Eq. 1:

$$Err = \sqrt{\sum_{i=1}^{N}(x_i - \hat{x_i})^2} \tag{1}$$

Recurrent neural network

In traditional neural network, the information is only propagated forward assuming that all inputs and outputs are independent. It is known as feedforward neural network (FNN). The idea behind recurrent neural network (RNN) is to make use of the outputs of the hidden layers as feedback connections to serve as inputs to the neurons. In other words, RNNs integrate

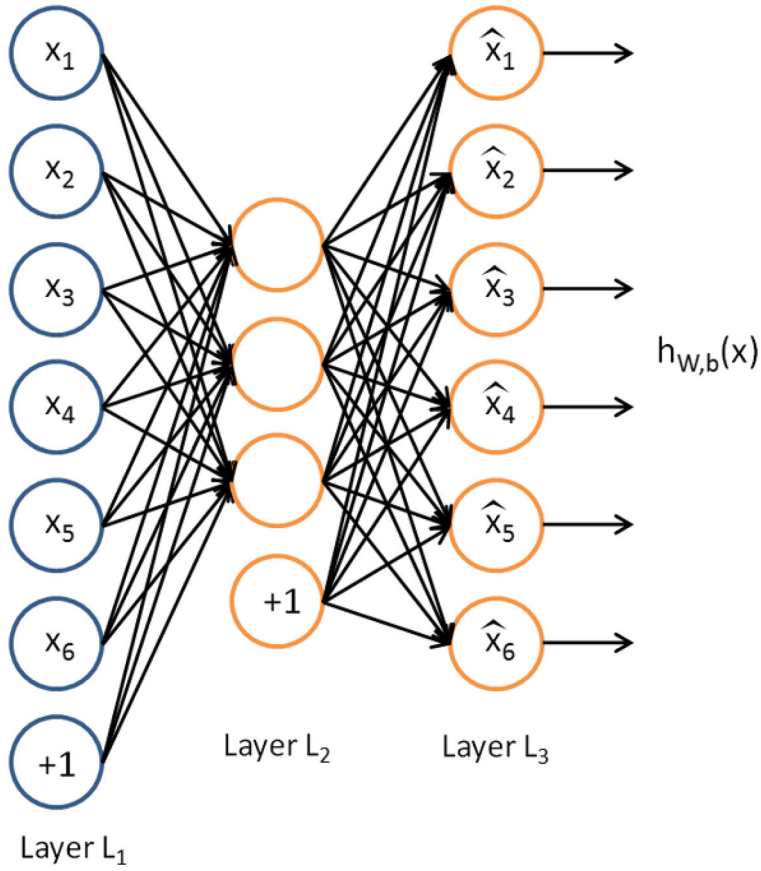

Fig. 2 Auto-Encoder (image from: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/)

the output from previous computations offering context information for applications based on sequential data. Figure 3 shows a recurrent neural network and the unfolding in time. x_t is the input at time t, and s_t is the memory of the system which is calculated using the previous state: $s_t = f(Ux_t + Ws_{t-1})$. f is a nonlinearity function, and o_t is the output a step t.

Many RNNs have been proposed, such as Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997). LSTM makes it easier to learn long-term dependencies in the input. It is composed of an input neuron, a recurrent hidden block whose basic unit holds a memory cell with an associated cell state and an output neuron. The hidden blocks are made of a set of recurrently connected sub-blocks. Each block is composed from one or more self-connected memory cells and from the following multiplicative units: the input, the output, and sigmoid layers called the forget gates. Through this, one can continuously write, read, and reset operations on the cells. These units allow the LSTM cells to memorize the data for long periods, thereby solving the vanishing gradient problem

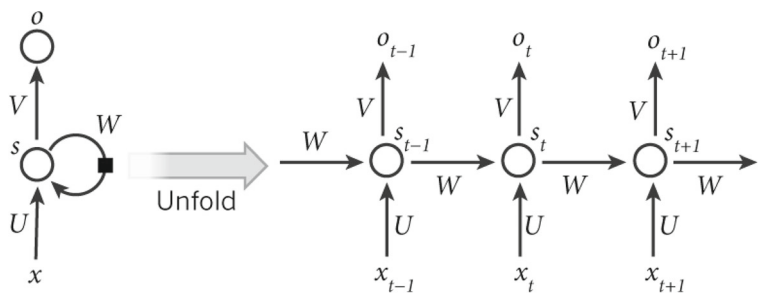

Fig. 3 A recurrent neural network (image from: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/)

present in classical RNNs. For example, if we assume that the input unit is closed, the new inputs won't be able to overwrite the activation of the cell. Therefore, the network can use this information later when the output gates are open.

K-means clustering

Clustering seeks to divide data into a finite number of clusters. The K-means clustering is surely the most well-known clustering algorithm: Given n data points $x_i, i = 1...n$, the goal is to attribute each data point to one of the K clusters by finding the positions $\mu_i, i = 1...k$, of the clusters that reduce the distance between the data point and the cluster:

$$argmin \sum_{i=1}^{K} \sum_{x \in c_i} d(x, \mu_i) = argmin \sum_{i=1}^{K} \sum_{x \in c_i} ||x - \mu_i||_2^2 \tag{2}$$

where c_i is the set of data points belonging to cluster i. K-means is very popular because it is very simple and converges rapidly. Algo. 1 shows the K-means algorithm.

Algorithm 1 K-means algorithm.

1. Initialize $\mu_1 ... \mu_K$ randomly
2. Assign each data point to the closest centroid: $c_i = j : d(x_j, \mu_i) \leq d(x_j, \mu_l), l \neq i, j = 1...n$
3. Recompute each μ_i as the centroids of all data points belonging to that cluster
4. Repeat steps 2–3 until the centroids no longer move.

This algorithm searches for a global minimum but converges to a local minimum since the problem is NP-hard and the proposed solution is heuristic.

Proposed method

System overview

The key idea in our approach is that the most recent observation should get a little more weight than older observations. In other words, the predicted value with some recent neighboring points should have more importance and weight in the process of choosing which scenario this actual behavior represents. An illustration of this idea is presented in Fig. 4. We assume that we are supplied with a training data set $X_i (i = 1, 2, ..., N)$, where all the provided data represent a normal behavior during the day. Our proposed method has two stages: training phase and testing phase. Before the training phase, we begin by generating 24 groups corresponding to the 24 h of the day, one group for each hour. For example the group number one corresponds to the hour 1:00 A.M. and has all the vectors of length 24 representing the power consumption data of the user for the 24 h preceding 1:00 A.M. Figure 5 presents the procedure used for the group selection phase. After re-generating the data, we applied the K-means algorithm to partition each group into k disjoint clusters $C_1, C_2, ..., C_k$. In the same time, we train a LSTM neural network which returns the predictions of the next data point using the last sequence of power consumption data. Let's say we are at time: $t = n - 1$, in order to predict the power consumption at time $t = n$, we take the last 23 power values and we use the trained LSTM to predict the next power consumption as shown in Fig. 6.

In fact, many of the user's behaviors are random but, due to the user's living habits, identical daily consumption behavior can appear repeatedly. By taking the power consumption data of a whole year as our training data, we assume that the majority of these random events have occurred and thus, they are included in our training data. Some events are completely random as turning on and off the coffee machine. Note that the trained model won't be affected by that since the power consumed by these events is negligible. Nevertheless, in case the user completely changes his habits (for example he started a new night shift job), then we have to reconstruct a new training data and to retrain the whole model. In the next two subsections, we are going to give the details of these two steps summarizing our proposed method.

Finding the K clusters

We are going to use the K-means clustering approach, and we're going to assume that the observed data represent the normal behavior. The K-means algorithm is used for clustering large amount of data. This algorithm is going to generate different classes of time series data by selecting the centroid representing all the instances in each cluster. We begin by splitting the

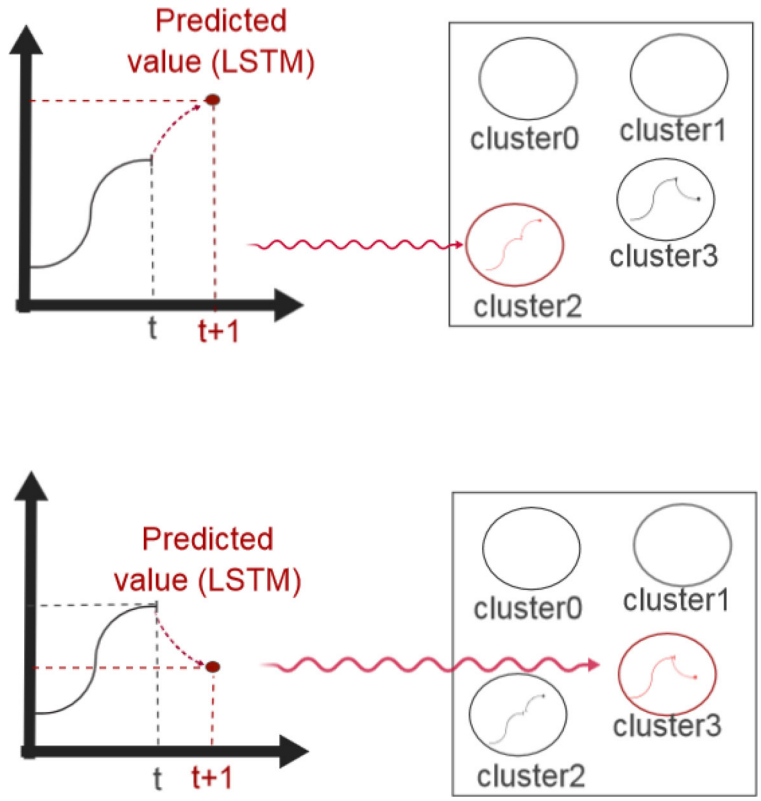

Fig. 4 Recent data weights more than old data

time series data into segments. The length of the segment is fixed to 24 and the beginning of each segment could be anywhere in the day.

The value K, the number of clusters, used for the K-means algorithm are 5, 10, 11, 20, and 30. We were unable to test the algorithm for larger k, since the

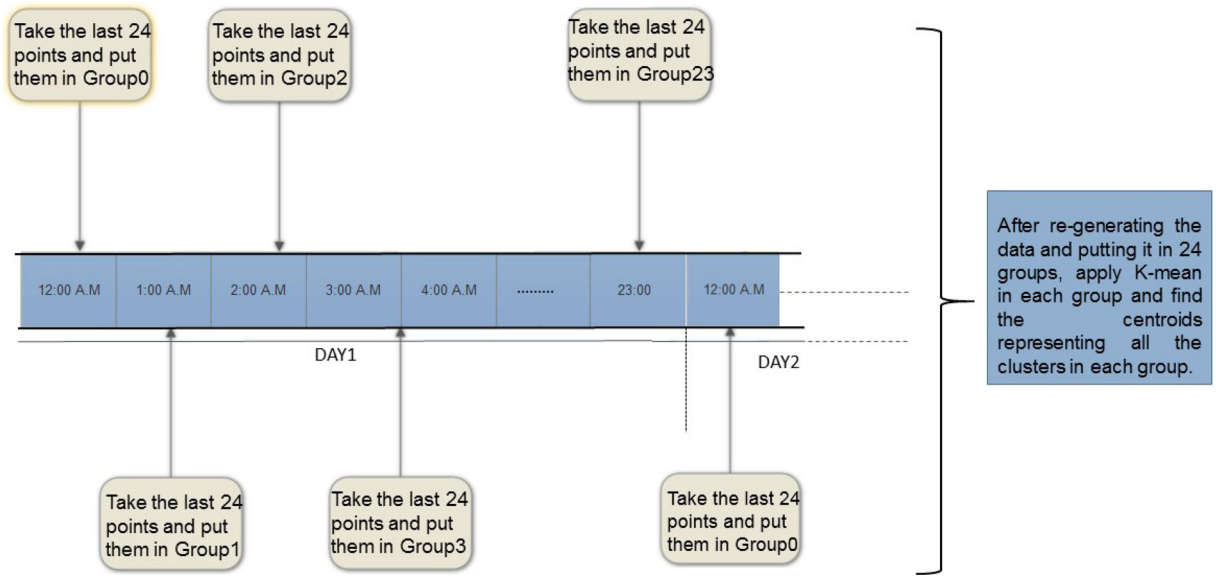

Fig. 5 Generating 24 groups corresponding to the 24 h of the day, one group for each hour

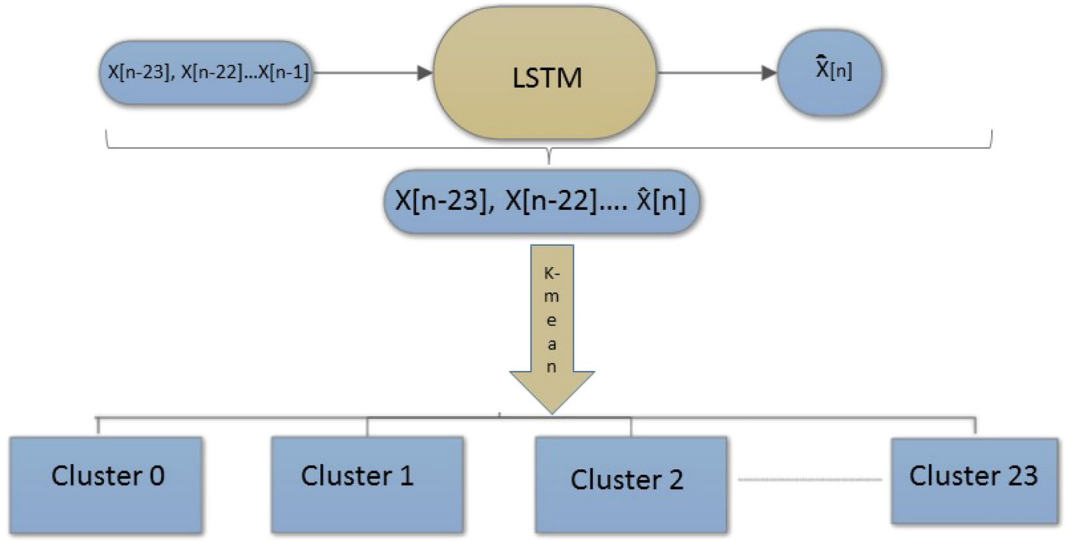

Fig. 6 MainMethod

algorithm failed to converge. The superior performance was attributed to $k = 11$. In theory, increasing K can produce better profiles but in our case the algorithm did not converge to an acceptable solution in the allowed time frame. Ideally, there are some algorithms like the BIC algorithm that may help choosing a more optimal K. Due to computational and practical considerations, we limited the search space for only five values and we believe that the value $k = 11$ produced meaningful centroids.

Predicting the next power consumption value

Prediction-based anomaly detection has been widely used in statistics community (Abraham and Chuang 1989; Abraham and Box 1979). The motivation behind these techniques is the assumption that an observed behavior is going to reoccur in the future (maybe with minor changes). Another key step in these techniques is the assumption that the data used for training describe well the process. In other words, they are considered as "normal." In this work, we trained a RNN, more specifically an LSTM, that returns the predictions of the next data point using the last sequence of power consumption data. Assuming that the model trained represents well the process, if the predicted values are far distant from the expected ones, we can say that the model detected an anomaly. So in fact the model has to compare the predicted value either with a predefined model or with the real data stream values to determine if the data point present is anomalous. The intuitive approach is to calculate the euclidean distance between the predicted data and the real data, and then to compare it with a predefined threshold.

Dataset and behavior analysis

In order to test the efficiency of the proposed method, several experiments are performed. This section describes the dataset we used in our experiments as well as an a general analysis of the user's behavior using the boxplot technique.

Dataset

The public dataset we used is called Dataport, and it is collected from Pecanstreet[1]. The main goal of Pecanstreet is to accelerate the research and the innovation in water and energy consumption. The full database contains data from smart meter readings of power utilization from 67 electronic devices in nearly 1000 houses. A data sample is shown in Fig. 7 where the "DataId" represents the id for each house, the "Time" represents the datetime of the measure and "use" represents the power consumption at that time of the day. In this work, we randomly chose to study

[1] https://dataport.pecanstreet.org/

	Time	DataId	use
0	2017-01-01 06:00:00	26	1.788817
1	2017-01-01 07:00:00	26	1.223917
2	2017-01-01 08:00:00	26	0.668367
3	2017-01-01 09:00:00	26	0.478667
4	2017-01-01 10:00:00	26	0.416283

Fig. 7 An example of power consumption data (in KW)

the behavior of one user during the period of January 2017 to December 2017. The user id number is 26 in this database.

Behavior analysis

In Fig. 8, we used a boxplot to illustrate the energy consumption of a user, taking 20 consecutive weekdays and 20 consecutive weekends. The boxplot is a common statistical technique used to identify hidden patterns through a graph approach. The box represents the interquartile range which is the part between the lower and the upper quartiles. The median is indicated by a line. Outliers show the data points lying between 1.5 and 3 box lengths. This has been considered as acceptable for most situations (Frigge et al. 1989). According to Fig. 8, the behavior differs between weekdays and weekends. The reason is that users tend to spend more time at home during the weekends; thus, they use more energy. Figure 9 presents a general behavior model of this user. Looking at the upper quartile (Q3) of the weekdays for example (the upper part of Fig. 9), we can see that there is an uprise in the power consumption at 6 o'clock, 18 o'clock, and 21 o'clock. This is explained by the fact that the occupants of this house get up in the morning around 6 o'clock and they are all back home around 18 o'clock.

Results and discussions

This section is divided into two subsections. In the first subsection, we try to find anomalies in a complete unsupervised scenario. Detecting anomalies is finding anomalous data points relative to some standard data. Unfortunately in our case, there is no information

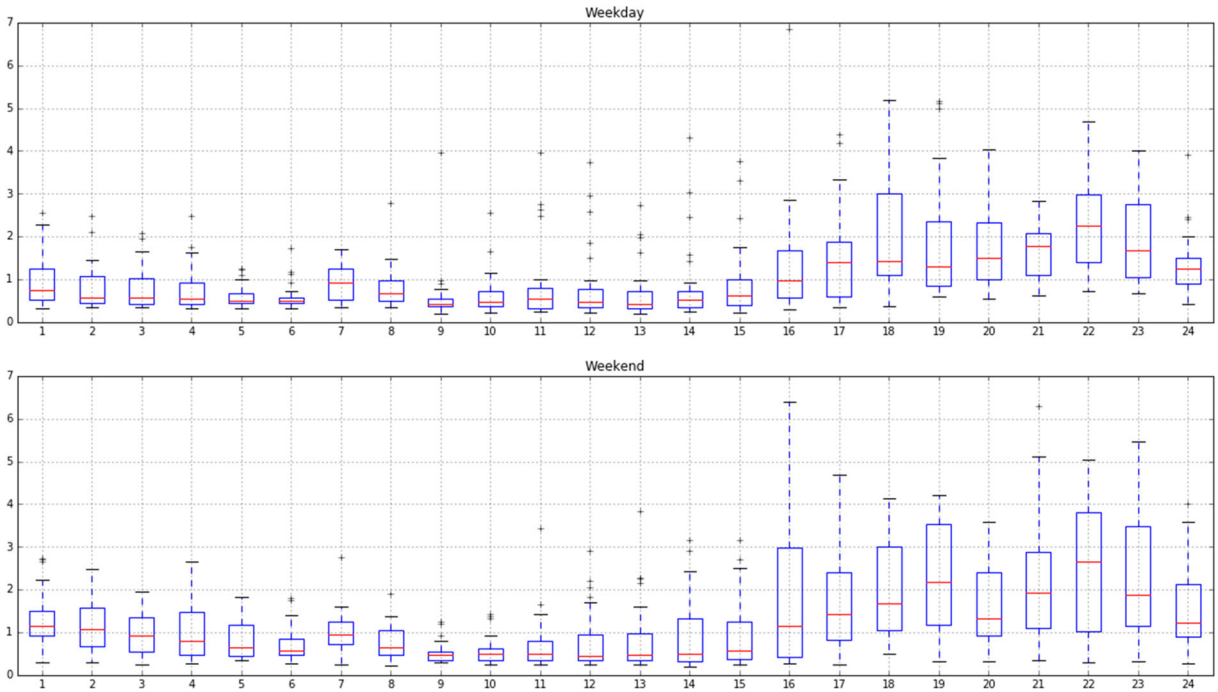

Fig. 8 General characteristics of data using boxplot

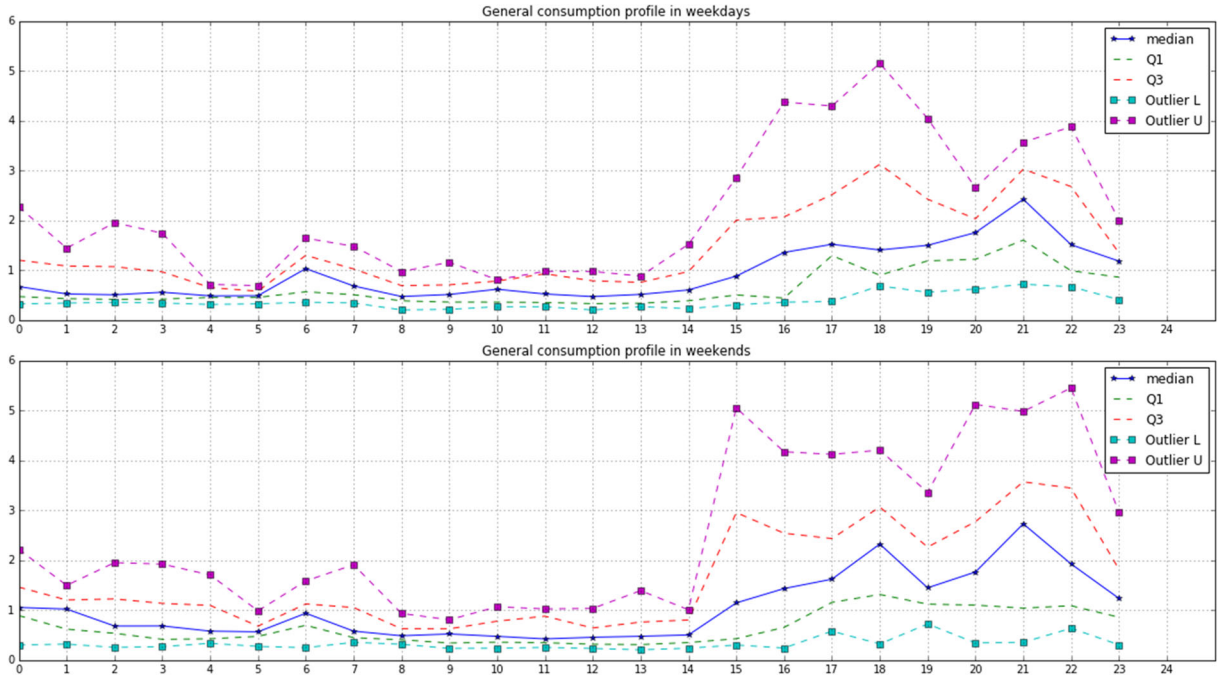

Fig. 9 A boxplot with the exact value for mean, lower quartile and outlier, and upper quartile and outlier

about which ones among the outliers are true anomalies or not (unsupervised data). Thus, in the first subsection, we couldn't adopt any performance measure and the results presented can only serve as an indicator to alert consumers by giving them assistance towards identifying faulty equipment or misconfigured machines consuming more energy than required.

In the second subsection, we manually insert random anomalies in order to provide performance measure. In other words, we insert outliers and we assume that these outliers are true anomalies.

Anomaly detection without inserting artificial anomalies

In this section, we present the results of our proposed approach combining the LSTM method with the *K*-means method and compare it with the individual *K*-means and LSTM methods applied separately. We begin by using Auto-Encoders in order to detect anomalous days without specifying at what time the anomaly has occurred. The goal is to see later if the days where our proposed method detects anomalies are the same as the days detected as anomalous by the Auto-Encoders.

Detecting anomalous days with Auto-Encoder

In this approach, we aim to find the anomalous days without localizing the anomaly in the time. An Auto-Encoder is a neural network trained to reconstruct the input data. That's why the output has the same size of the input layer, while one of the hidden layers is smaller than the input layer and it is called the bottleneck layer. In this work, the Auto-Encoder used is made from the following layers (24, 50, 20, 2, 20, 50, 24). We have 24 neurons for each of the input and the output layer (the first and the last elements of the vector). The number of hidden layers used in our network is five (the number of neurons in each of the five hidden layers is respectively 50, 20, 2, 20, and 50). Once trained, the hidden layers will learn a feature detection architecture. The bottleneck will force the network to represent the data in a compact representation.

Using the mean square error (MSE) as an objection function, an error of 0.017 can be obtained in nearly 30 epochs. The experimental results have shown that a similar error permits the network to generalize well and to be able to reproduce all the data that have small variation with respect to training data. We built two different models treating the days depending if they

do belong to weekdays or weekends. For the models with two dimensions, we are able to visualize our data. The result of visualization for the Auto-Encoder with 5 hidden layers is shown in Fig. 10 for weekdays and Fig. 11 for weekends. Each of these colors represents the reconstruction MSE of each point representing each of the nine test days. The reconstruction error, which is the error between the original data and its new representation, is used as an anomaly score. A threshold can be set on the reconstruction error to detect anomalies. If the reconstruction error is higher than a given threshold, it means that the test day is very different from the days in the training set considered as normal. We set the threshold at 0.04 for weekdays and 0.06 for weekends. The thresholds used were selected by observing the train dataset. In other words, the model is capable to reproduce the majority of the training dataset with an error smaller than 0.04 for weekdays of the training and 0.06 for weekends of the training. The threshold of the weekdays is smaller because apparently the occupant tends to have a more obvious daily routine during weekdays. This can be explained by the fact that, for example, the occupant of this house always travels at the same time to and from work (or school).

For weekdays, the reconstruction error for days 4, 5, and 7 was higher than the defined threshold, and for weekends only day 2 has been considered as anomalous. The actual power consumption values of these test days are presented in the lower part of Figs. 10 and 11.

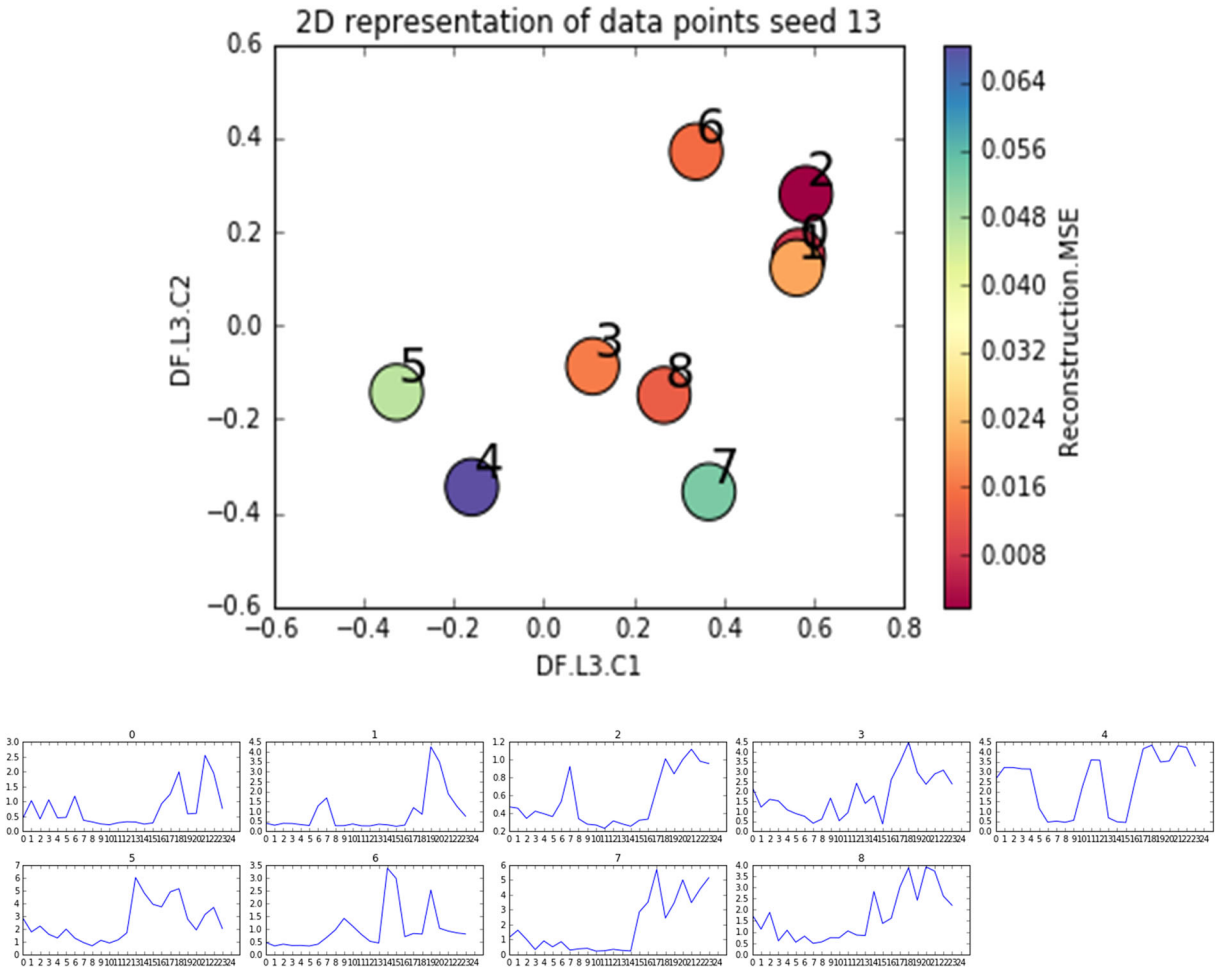

Fig. 10 In the top part, we visualize different test days belonging to weekdays using Auto-Encoders. Different colors represent the reconstruction MSE. In the lower part, the actual power consumption values of these test days are presented

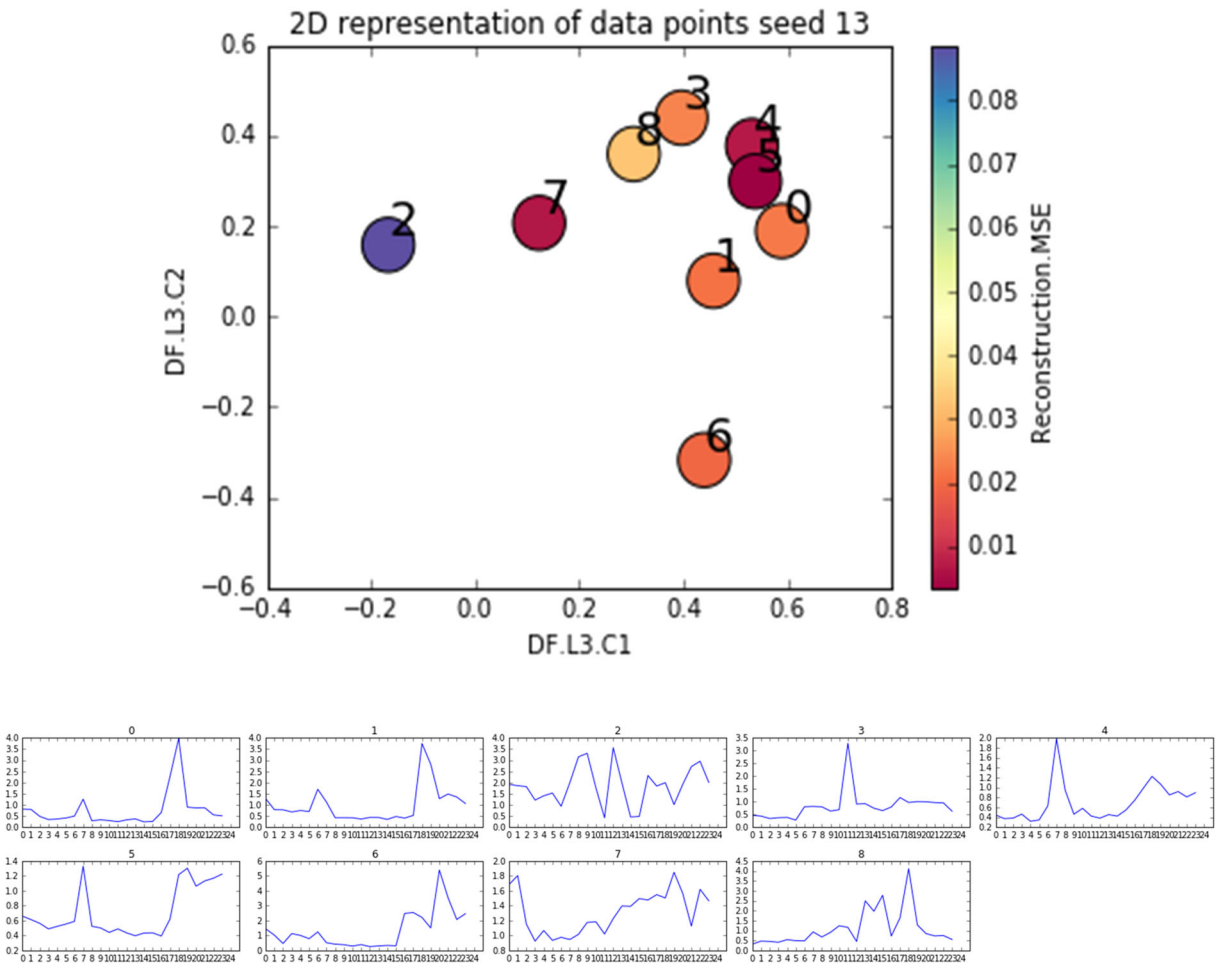

Fig. 11 In the top part, we visualize different test days belonging to weekends using Auto-Encoders. Different colors represent the reconstruction MSE. In the lower part, the actual power consumption values of these test days are presented

Local anomaly detection using our proposed method

Figures 12, 13, 14, and 15 illustrate the performance of our method which combines the K-means and the LSTM methods. The value of K for the clustering was set to 11. This choice was adopted after five trials (5, 10, 11, 20, 30). For the LSTM, the batch size was set to 64 and we stopped the training after 200 epochs using the early-stop technique. The optimizer used was Adam optimizer and the learning rate was set to 0.0001. The model was evaluated with three stacked LSTM layers of sizes 32, 64, and 100, with a standard dropout on the output of each layer equal to 0.2. When the predicted value is higher than the measured one by a threshold margin, an anomaly is detected. The threshold chosen was set to $T = 6$ and the error was calculated using the MSE. Note that when the Y values of the anomalies (the red circles in the Figure) are equal to zero, it means there is no anomaly.

The results show that the combination of the LSTM with K-means succeeded to predict values that are most of the time consistent with the curve representing the real power consumption data. However, some test days were very different from the learning database and thus, the predicted values were deviated. For example, the test day number 4 that belongs to the weekdays (see Fig. 12) is so deviated from the predicted curve even though no anomalies have been detected (the error is below the threshold). This can explain why this day has been considered anomalous by the Auto-Encoder (see Fig. 10), since the overall behavior of this day is not a common profile. Which

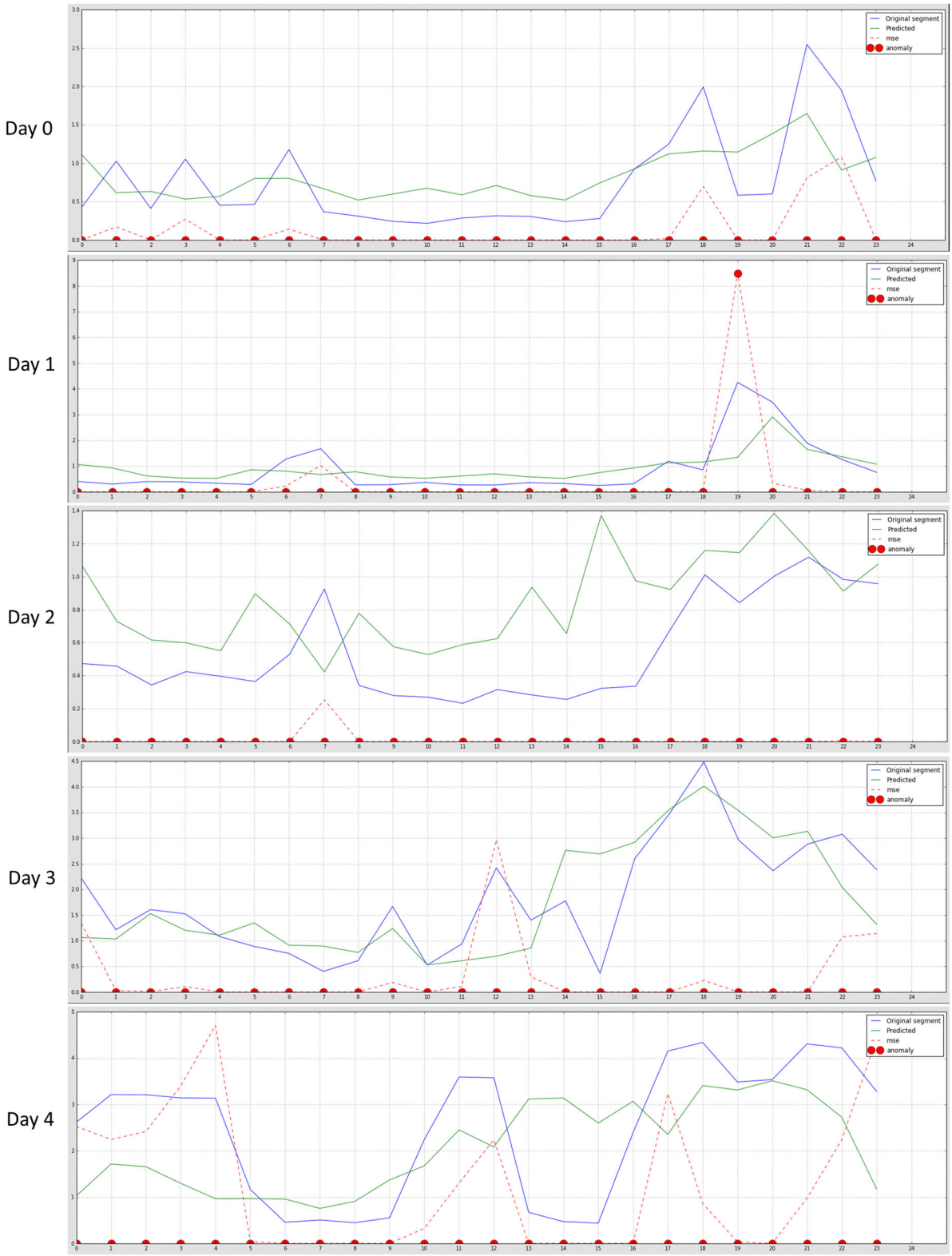

Fig. 12 Power consumption values for five test weekdays (day 0 to day 4). The actual power consumption, the predicted one, and the anomalies are presented

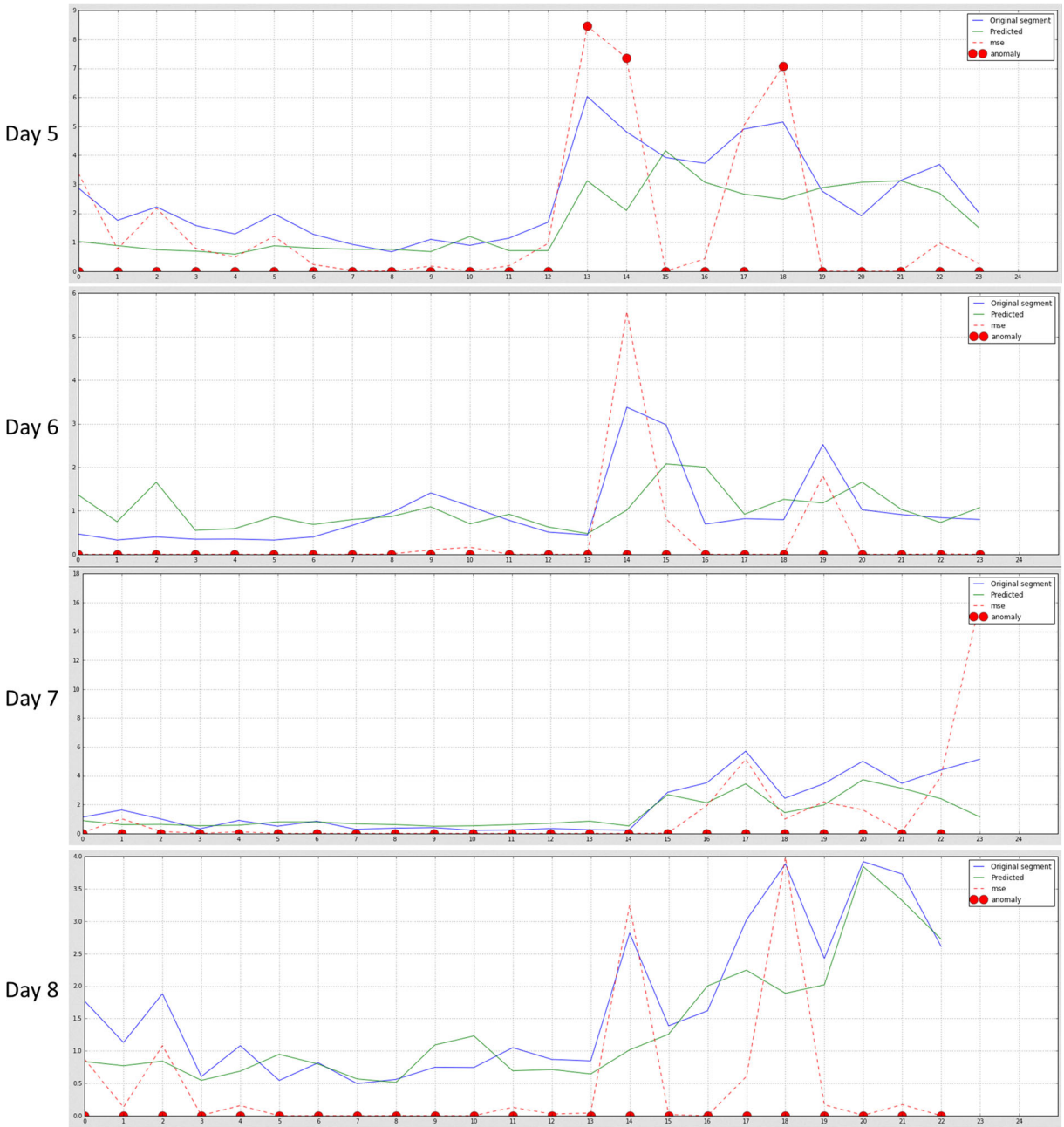

Fig. 13 Power consumption values for four test weekdays (day 5 to day 8). The actual power consumption, the predicted one, and the anomalies are presented

means, while Auto-Encoders judge the overall profile of the day, our proposed method tries to localize the anomaly and does not consider any uncommon behavior as an anomaly. Contrary to the Auto-Encoder algorithm that considered the day number 1 in weekdays as a normal day, the proposed algorithm detected an anomaly at 19:00 P.M. where the power consumption exceeded 4 KWH while the model has predicted a value of 1.3 KWH.

Similar analysis can be applied to the rest of the days, noting that weekdays numbers 5 and 7 were considered as anomalous by the Auto-Encoders (threshold

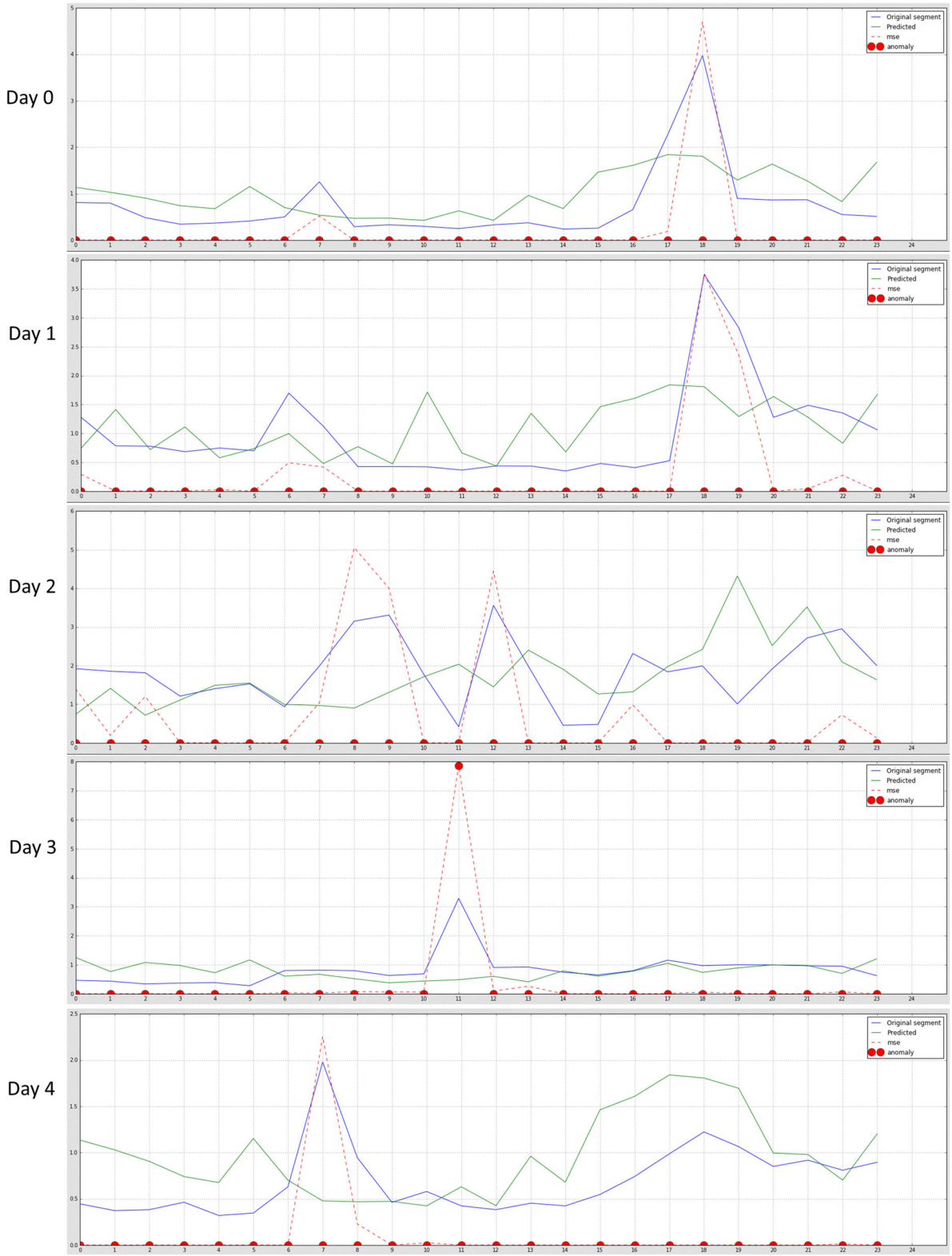

Fig. 14 Power consumption values for five test weekends (day 0 to day 4). The actual power consumption, the predicted one, and the anomalies are presented

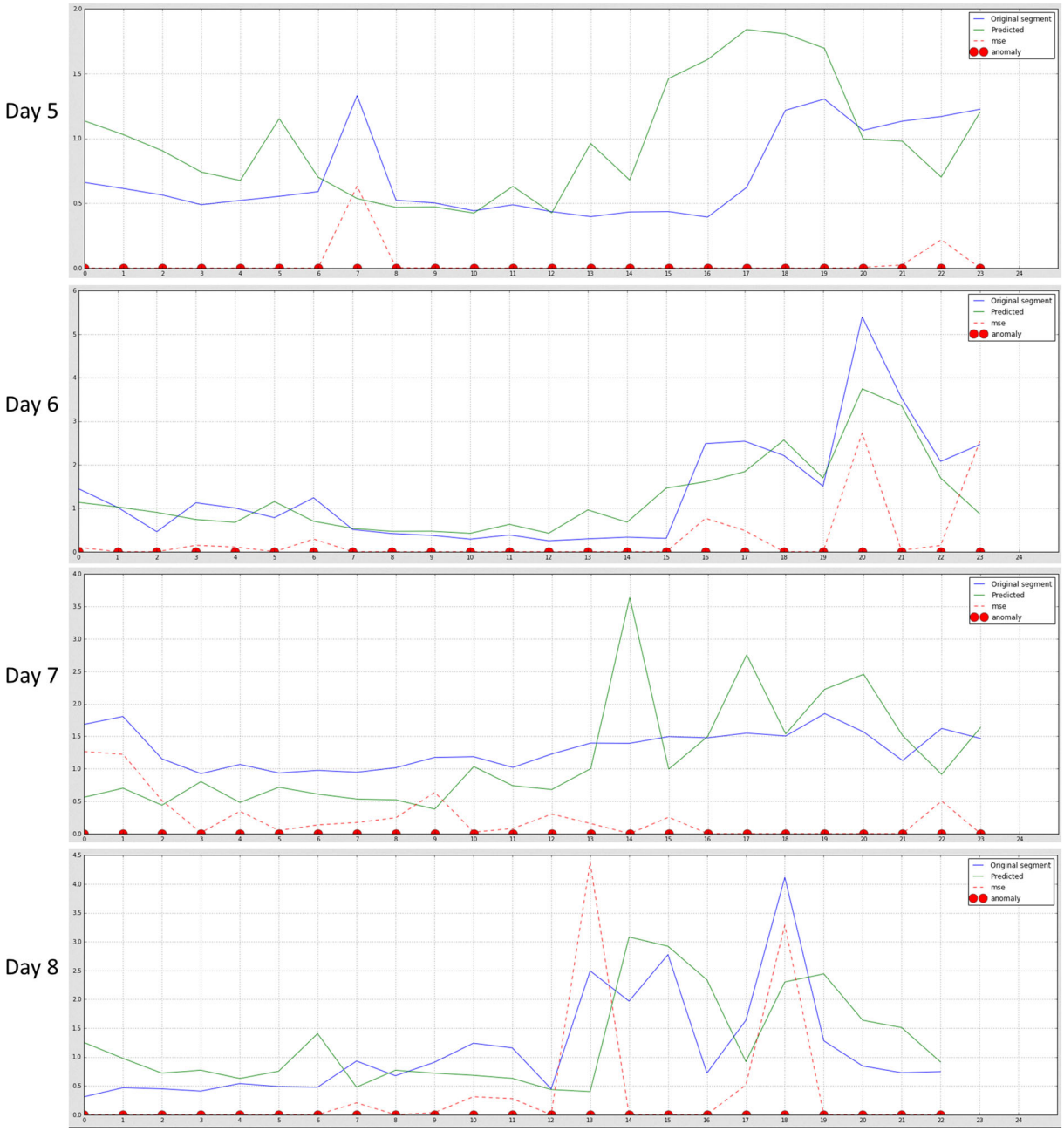

Fig. 15 Power consumption values for four test weekends (day 5 to day 8). The actual power consumption, the predicted one, and the anomalies are presented

= 0.04). Our proposed method confirmed that and localized the anomaly in the time (13:00 P.M. and 14:00 P.M. for day 5 and 23:00 P.M. for day 7). Tests performed on weekends (Figs. 14 and 15) showed a decrease of the prediction capability of the proposed algorithm since the training days belonging to weekends are fewer. The only anomaly detected in weekends test days is in day 3 at 11:00 A.M. where the real consumption data was higher than 3 KWH whereas the predicted data was 0.5 KWH.

Comparison, training cost, and computational speed

Figure 16 presents an example to compare between the proposed methods, K-means alone and LSTM alone. It shows that the combination of K-means with LSTM, as we have proposed, gives results more close to the real power consumption data than that of the single K-means or the single LSTM. Compared between the 3 plots in Fig. 16, our proposed method is significantly lower than the two others in the error rate. In fact, the results show that between 12:00 A.M. and 6:00 A.M., the 3 methods are mostly consistent with the curve representing the real power consumption data (error<0.035). For the last part of the curve (after 16:00 P.M.), the K-means method and the LSTM method are clearly separated from the original segment, while our proposed method conserves a better overall prediction specially at 19:00 P.M. and 21:00 P.M.

The proposed approach consists of a training phase and a testing phase. In terms of training time, LSTM needs about 80.6 min to train a model on a Quadro K610M GPU and 0.24 s for the test phase. Concerning the K-means algorithm, it needs around 11.25 s on a 2.8-Ghz quad core CPU and 0.06 s for the testing phase. This algorithm utilizes iterative refinements until the centroids stop from moving. The complexity of this algorithm is linear and it depends on the number of data points. If we apply the K-means on N data samples, the complexity would be $O(k.N)$. The authors of Soheily-Khah et al. (2016) provided a proof

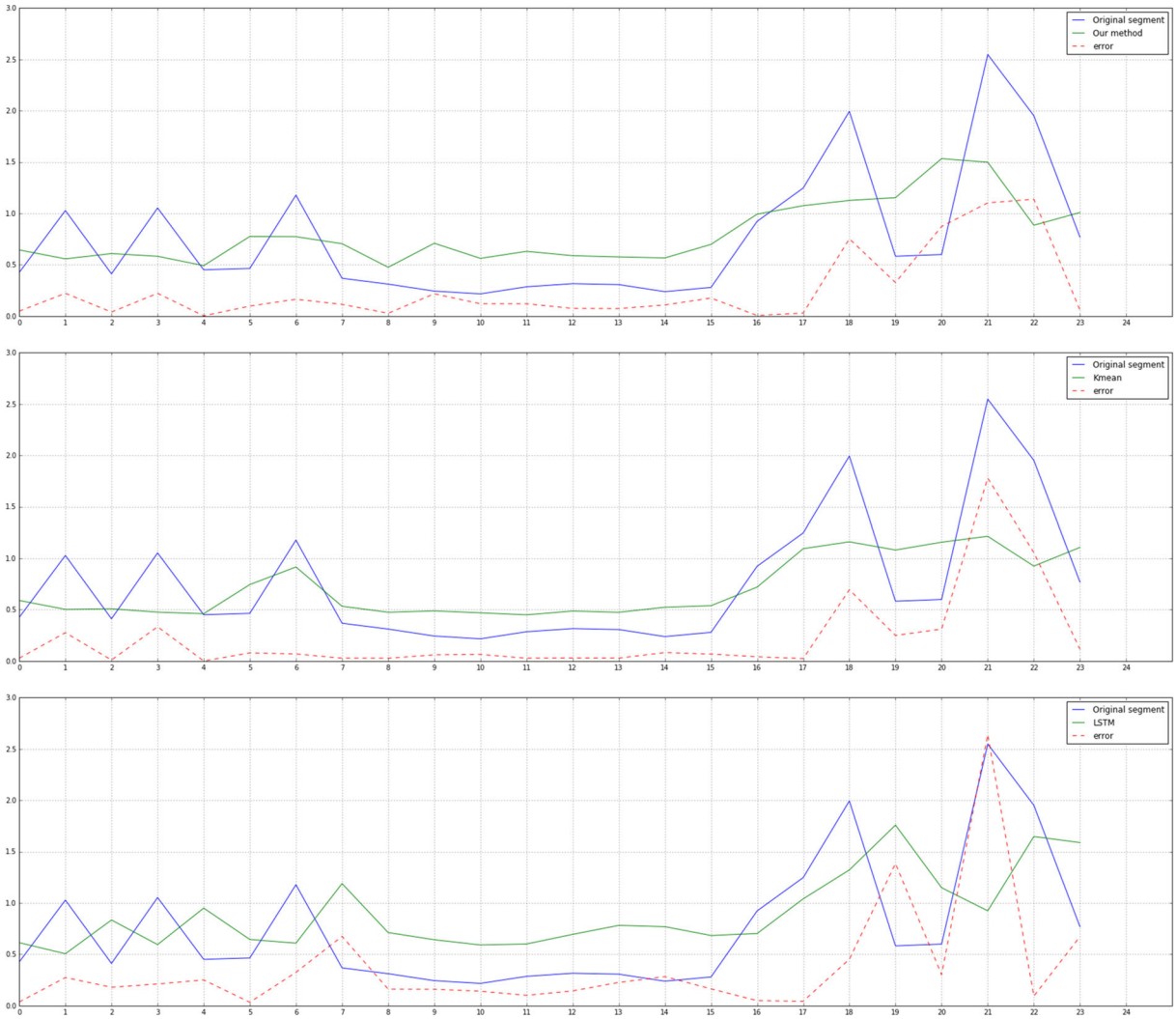

Fig. 16 The power consumption data predicted using the proposed method, K-means alone, and LSTM alone

of the finite convergence of the K-means algorithm. However, in some particular cases, it fails to converge to a local minimum. On the other hand, the complexity of the methods relying on LSTM increases with the increase of the number of layers. In fact, the computational complexity of learning LSTM models per weight and time step is $O(1)$. Thus, assuming W is the total number of parameters in the network, the corresponding computational complexity per time step would be $O(W)$. These deep learning algorithms are time consuming and require powerful hardware for training besides a large number of training data.

Performance measure by inserting labeled anomalies

In this part, we are going to manually insert outliers randomly in our test data. We are also going to assume that all the inserted outliers are true anomalies. The pseudo-code of the algorithm we used to insert anomalies is represented in Algo. 2 and it was inspired from the work presented in Jokar et al. (2015). Among the 6240 samples of the original dataset, we used 6000 samples for training (250 days) and 240 samples for testing (10 days). We then randomly generate 40 anomalies, 4 anomalies per test day.

Algorithm 2 Inserting malicious samples.

Input: h(t): original data
Output: h1(t): original data with malicious samples

- $NbAnomaliesPerDay = 4$
- Beta=2
- h1(t)=h(t)
- for i in range ($NbAnomaliesPerDay$) :
  location= random(0,24) # any time of the day
  h1(location)= Beta*h(location)

Evaluation metrics

Table 1 summarizes the confusion matrix we used in order to evaluate the performance of our algorithm. Equations 3, 4, and 5 show the definitions of accuracy, precision, and recall respectively. The accuracy reflects the proportion of the correct predictions to all the samples. Precision shows the ability of the algorithm to identify positive and negative data whereas recall shows the ability to detect all the positive samples.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$precision = \frac{TP}{TP + FP} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

Results and discussion

Figure 17 provides case study of of our method (LSTM+K-means) and three state of the art methods: Auto-Regression (AR), Auto-regressive Integrated Moving Average (ARIMA), and Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors model (SARIMAX). This test has been performed on the day number 1 of the Week-Days test. We can notice that the AR method detects only two out of 4 anomalies and it has one false-positive detection. Meanwhile, ARIMA and SARIMAX also detect two out of four anomalies but they don't have any false-positive detection. Finally, our method detects three out of four anomalies and it has one false-positive detection.

To demonstrate more convincing results, the same experiment has been repeated on all test days and the results are reported in Table 2. We can notice that the accuracy of the AR method is the lowest and the results contain the highest false-positive rate. Meanwhile, the accuracy of ARIMA is better and it has less false-positive samples but still the recall is the lowest. Our method and SARIMAX yield the best recall score (0.80) while our method has a better accuracy (0.89) and a better F1 score (0.71). Note that while false-positives are bad, they are not the worst outcome. A false positive can generate a false alarm and can cause waste of time in the monitoring system. The worst scenario is having a high false-negative rate. In

Table 1 Confusion matrix

	Positive	Negative
True	TP: an anomaly sample predicted correctly	TN: a normal point predicted correctly
False	FP: a normal point not predicted correctly	FN: an anomaly sample not predicted correctly

Fig. 17 The red shaded regions represent anomaly periods. Red stars represent the inserted anomalies. Red circles represent the detected anomalies. The red dash line is the mean square error between the predicted power consumption and the real power consumption

other words, it is the case where a problem is occurring but the monitoring team is not aware of this problem. Moreover, our method yields a RMSE of 1.032 where for all other methods this value lays between 1.52 and 2.07. Which means, our proposed method is better in predicting the power consumption and in detecting anomalies.

Table 2 Comparison with some state of the art methods

Method	Accuracy	Precision	Recall	F1	RMSE
Auto-Regression (AR)	0.78	0.417	0.77	0.54	2.07
ARIMA	0.80	0.45	0.72	0.553	1.52
SARIMAX	0.79	0.43	0.80	0.559	2.03
Our method	0.89	0.65	0.80	0.71	1.032

Conclusion and future work

Analyzing unexpected patterns in time series data is a very challenging problem. In this paper, we developed the LSTM+*K*-means anomaly detection method. Our method is based on cascading two famous machine learning approaches: the LSTM and the *K*-means. First, the *K*-means algorithm is applied for each of the 24 groups representing the 24 h of the day in order to partition each group of data into *k* clusters. After learning the clusters within each group, a LSTM is trained in order to predict the power consumption of the next hour. The predicted value is concatenated with previous data to find the corresponding centroid in the corresponding group. We also used Auto-Encoders to detect anomalous days without specifying at what time the anomaly has occurred. Results are compared with other state of the art methods showing the superiority of the proposed method in terms of higher accuracy and better anomaly detection rate.

References

Energy Renewable (2010). Energy efficiency trends in residential and commercial buildings.

UNEP (2017). UNEP, United Nations Environment Program. https://www.tandfonline.com/doi/full/10.1080/09613218.2017.1356127.

Abraham, B., & Box, G.E.P. (1979). Bayesian analysis of some outlier problems in time series. *Biometrika*, *66*(2), 229–236. 10.1093/biomet/66.2.229.

Abraham, B., & Chuang, A. (1989). Outlier detection and time series modeling. *Technometrics*, *31*(2), 241–248. https://doi.org/10.1080/00401706.1989.10488517. https://amstat.tandfonline.com/doi/abs/10.1080/00401706.1989.10488517.

Antmann, P. (2009). Reducing technical and non-technical losses in the power sector. World Bank. http://documents.worldbank.org/curated/en/829751468326689826.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends® in Machine Learning*, *2*(1), 1–127. ISSN 1935-8237. https://doi.org/10.1561/2200000006.

Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M. (2015). Time series analysis: forecasting and control. John Wiley & Sons.

Chandola, V., Banerjee, A., Kumar, V. (2009). Anomaly detection: a survey. *ACM Comput. Surv.*, *41*(3), 15:1–15:58. ISSN 0360-0300. https://doi.org/10.1145/1541880.1541882.

Chou, J.-S., & Telaga, A.S. (2014). Real-time detection of anomalous power consumption. *Renewable and Sustainable Energy Reviews*, *33*, 400–411. ISSN 1364-0321. https://doi.org/10.1016/j.rser.2014.01.088. http://www.sciencedirect.com/science/article/pii/S1364032114001142.

Frigge, M., Hoaglin, D.C., Iglewicz, B. (1989). Some implementations of the boxplot. *The American Statistician*, *43*(1), 50–54. https://doi.org/10.1080/00031305.1989.10475612. https://www.tandfonline.com/doi/abs/10.1080/00031305.1989.10475612.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Jokar, P., Arianpoo, N., Leung, V.C. (2015). Electricity theft detection in ami using customers' consumption patterns. *IEEE Transactions on Smart Grid*, *7*(1), 216–226.

Lee, W., Stolfo, S.J., Chan, P.K., Eskin, E., Fan, W., Miller, M., Hershkop, S., Zhang, J. (2001). Real time data mining-based intrusion detection. In *DARPA Information Survivability Conference & Exposition II, 2001. DISCEX '01. Proceedings*, (Vol. 1 pp. 89–100). https://doi.org/10.1109/DISCEX.2001.932195.

Liu, F., Jiang, H., Lee, Y.M., Snowdon, J.L., Bobker, M. (2011). Statistical modeling for anomaly detection forecasting and root cause analysis of energy consumption for a portfolio of buildings.

Nadai, M.D., & van Someren, M. (2015). Short-term anomaly detection in gas consumption through ARIMA and artificial neural network forecast. In: 2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings, pp 250–255. https://doi.org/10.1109/EESMS.2015.7175886.

Soheily-Khah, S., Douzal-Chouakria, A., Gaussier, E. (2016). Generalized k-means-based clustering for temporal data under weighted and kernel time warp. *Pattern Recognition Letters*, *75*, 63–69. ISSN 0167-8655. https://doi.org/10.1016/j.patrec.2016.03.007. http://www.sciencedirect.com/science/article/pii/S0167865516000763.

Zhang, Y., Chen, W., Black, J. (2011). Anomaly detection in premise energy consumption data. In *2011 IEEE Power and Energy Society General Meeting* (pp. 1–8). https://doi.org/10.1109/PES.2011.6039858.