Journal of King Saud University – Computer and Information Sciences 31 (2019) 392–402

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

An adaptive framework for real-time data reduction in AMI

Marwa F. Mohamed [a,*], Abd El-Rahman Shabayek [a,b], Mahmoud El-Gayyar [a], Hamed Nassar [a]

[a] Computer Sciences Department, Faculty of Computers and Informatics, Suez Canal University, Ismailia, Egypt
[b] Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg

ARTICLE INFO

Article history:
Received 5 September 2017
Revised 27 February 2018
Accepted 27 February 2018
Available online 6 March 2018

Keywords:
Real-time data reduction
Forecasting methods
Advanced Metering Infrastructure (AMI)
Decision tree algorithms
Cloud

ABSTRACT

In existing Advanced Metering Infrastructure (AMI), data collection intervals for each smart meter (SM) typically vary from 15 to 60 min. If we have 1 million SMs that transmit data every 15 min, these SMs will export 4 million records per hour. This leads to dramatically increasing bandwidth usage, energy consumption, traffic cost and I/O congestion. In this work, we present an adaptive framework for minimizing the amount of data transfer from SMs. The reduction in the framework is forecasting-based; when an SM reading is close to the forecasted value, the SM does not transmit the reading. In order for the framework to be adaptive to the ever-changing pattern of SM data, it is provided with a pool of forecasting methods. A supervised-learning scheme is employed to switch in real-time to the forecasting method most suitable to the current data pattern. The experimental results demonstrate that the proposed framework achieves data reduction rates up to 98% with accuracy 96%, depending on the operational parameters of the framework and consumer behavior (statistical features of SM data).

1. Introduction

Advanced Metering Infrastructure (AMI) is one of the most important achievements in smart grid (Kabalci, 2016). The main goal of AMI is to collect energy consumption and power quality from smart meters (SMs) and to transmit them to a utility center for storage and further analysis. In general, The AMI involves several components connected through a network: SMs, aggregators, meter data management system (MDMS). The AMI has two different architectures: direct (without aggregators) and indirect (with aggregators) (Ghasempour, 2015). In a direct architecture, the data is transferred directly from SMs to MDMS in the cloud. On the other hand, the indirect AMI architecture includes aggregators between cloud and SMs; the aggregator collects the data from SMs, and propagates it through the network core to the MDMS.

In existing AMIs, the data collection intervals can vary from 15 to 60 min that causes the AMI data to grow very quickly (Alahakoon and Yu, 2015). For example, Austin Energy in Texas has implemented 50,000 SMs that transmit data every 15 min (Yan and Su, 2016) that produce around 4,8 million records daily. This may cause network bottlenecks problems. Real-time data reduction can be used to (i) minimize network bandwidth, (ii) minimize network energy consumption, (iii) increase I/O throughput, and (iv) reduce cloud storage and traffic costs (especially if pay-per-volume or pay-per-connection schemes are applied) (Papageorgiou et al., 2015).

Data reduction techniques can be classified into three types: data forecasting, data compression and in-network processing (Anastasi et al., 2009). In data forecasting techniques, a forecasting method is maintained in both the SM and aggregator/cloud layers. The method forecasts the energy consumption value and compares it to the actual value if the difference is within a predetermined error bound, the SM does not need to send the consumption value to the aggregator/cloud. In data compression techniques, the data is encoded in a reduced form while in network processing techniques, data is processed in intermediate aggregators; then the aggregators transmit the processed data rather than the raw data.

Data reduction techniques are widely used in wireless sensor networks (WSN) and IoT platforms for energy saving and communication bandwidth reduction. Therefore the framework presented in this paper, together with the ancillary methodologies are applicable there. Few works have applied these techniques in AMI. In Zeinali and Thompson (2016) used two compression schemes; Adaptive Huffman(AH) scheme is used to compress data at SM

[*] Corresponding author.
E-mail addresses: marwa_fikry@ci.suez.edu.eg (M.F. Mohamed), abdelrahman.shabayek@uni.lu (A.E.-R. Shabayek), elgayyar@ci.suez.edu.eg (M. El-Gayyar), nassar@ci.suez.edu.eg (H. Nassar).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2018.02.012
1319-1578/

layer and Lempel-Ziv Welsh (LZW) schema is used to compress data at aggregators layer. In Foreman and Pacheco (2016) applied in-network processing; the aggregators are responsible for not only collecting data from SMs but also analyzing, extracting information from collected data and transferring them to the utility server. However these works do not support real-time data reduction. They are based on a posterior data reduction, i.e., once SM data are aggregated. In Carvalho et al. (2014) applied forecasting based data reduction by using adaptive simple linear regression (SLR) in AMI. However, the SM data is nonlinear, as shown in Section 5, and SLR is not efficient enough to capture non-linearity of these data.

Thus, this paper presents an x-layered framework that aims to 1) reduce the amount of data transferred between SMs and either the cloud (in direct AMI) or aggregators (in indirect AMI) in real-time, and 2) be efficient to capture data nonlinearities. Our framework is based on forecasting-based data reduction techniques by using light weight forecasting methods (naïve, moving average and exponential smoothing). We selected these methods as they require less memory and provide real-time computation.

The main challenges to face are 1) how to select an appropriate forecasting method for each consumer and 2) how to adaptively change the currently used forecasting method if the consumer behavior changes. The proposed framework robustly solve these issues as explained in Section 3 and 4.

The rest of the paper is organized as follows. Background and related work are covered in Section 2. Then a full description of the suggested adaptive framework is given in Section 3. Section 4 explains the workflow of the proposed framework. The experimental results are discussed in Section 5. Section 6 contains the discussions on related issues and future work. Finally, conclusions are presented in Section 7.

2. Background and related work

This section reviews related work in forecasting-based data reduction.

2.1. Forecasting-based data reduction

There are three main categories of forecasting-based data reduction: time series methods, regression methods, and machine learning techniques (Dias et al., 2016). Time series methods include naïve, autoregressive (AR), moving average, exponential smoothing, and autoregressive integrated moving average (ARIMA) methods. Regression methods include linear regression, kernel regression, and principal component analysis (PCA). Machine learning methods are based on artificial neural networks (ANNs) to reduce the frequency of data transmission. In this paper, the first category (time series methods) is used as they (i) require neither external data nor extended analysis to make accurate forecasting (independent methods), (ii) require less memory, (iii) provide real-time computation which is mandatory as the data reduction occurs in real-time, and (iv) capture nonlinearities data efficiently.

Selecting an appropriate forecasting method can depend on statistical features and/or physical features. Wang et al. are designing a meta-learning framework for selecting the most appropriate forecasting method for time series data depending on its characteristics (e.g. trend, seasonality, autocorrelation, skewness, kurtosis, non-linearity, self-similarity, and chaos) (Wang et al., 2009). Rasanen and Kolehmainen proposed clustering approach for creating accurate load energy consumption curve for customers based on statistical characteristics (e.g. average, standard deviation, skewness, kurtosis, chaos, and energy feature) (Rasanen and Kolehmainen, 2009). Cui et al. use both statistical and physical features (e.g. a number of stories, Area (m2), roof type, wall type, window type, cooling, and space type) for selecting an appropriate forecasting method for each unique building (Cui et al., 2016). In this paper, the selection of forecasting method depends on statistical features as in most developing countries some physical features have no real effect, in general, energy consumption as home size or the number of family members where a very small home may contain large family and vice versa. The statistical features used in this paper are trend, seasonality, periodicity, autocorrelation, skewness, kurtosis, non-linearity, self-similarity, chaos (Wang et al., 2009), average and standard deviation (Rasanen and Kolehmainen, 2009). Combining these features has shown improved selection accuracy as will be shown in Section 5.

In this paper, three decision tree algorithms, C4.5, CART and Hoeffding tree, are used to find the relation between light weight forecasting methods (N, MA and ES) and statistical features of consumption data (trend, seasonality, auto-correlation, skewness, kurtosis, non-linearity, self-similarity, chaos, average and standard deviation). C4.5 is a greedy divide and conquer algorithm that is used for classification and rule induction (Wu et al., 2008). It is considered as one of the most common and effective classification algorithms (Chauhan and Chauhan, 2013). Classification and Regression Trees (CART) is non-parametric and is used for classification or regression trees, depending on whether the response variable is continuous or categorical (Chen et al., 2015). A Hoeffding tree (VFDT) is an incremental decision tree algorithm that able to learn from massive amount data streams (Hulten et al., 2001). A comparison between different decision tree construction algorithms has been reported, then the algorithm with the best accuracy is employed as explained in Section 4.

2.2. Related work

Recently, some work has been done on real-time data reduction in AMI, IoT, and WSN. Carvalho et al. (2014) aim to reduce network traffic and minimize the used bandwidth of AMI data by using adaptive simple linear regression (SLR) approach. SM sends function coefficients of SLR method to the cloud instead of sending actual consumption data. If the error between actual value and forecasted value exceeds a certain threshold, the SM re-computes the function coefficients of SLR and re-sends them to the cloud. The main drawback of this approach is that consumer behaviors are variant over time. This may force SM to compute the function coefficients several times. Another drawback is that SLR is less efficient to capture nonlinearities in the actual consumption data.

Kang et al. (2016) aim to reduce the battery power and minimize the used bandwidth of body sensor data (heart rate and body temperature) by applying beacon data points after the inference of data processing. They are comparing the sensed data with adjacent data before and after. If the absolute difference between the sensed data, previous time step data, and next time step data exceeds the error threshold (variance rate), then the sensed data will be transmitted to the cloud. They select beacon data points which are always transmitted to the cloud at fixed intervals to enhance the reduction accuracy. The main drawbacks are 1) the delay of waiting time for the next sensed value to be compared, 2) the permanent transmission of beacon data points to the cloud which can be similar to the forecasted value.

The work proposed in Arbi et al. (2017) aims to reduce energy consumption in WSN by using self-exciting threshold autoregressive (SETAR) model. A dual forecasting model is applied in both sensor and data aggregation layer when the difference between the forecasted and actual value is greater than the threshold, the sensor transmits the actual value. The main drawback of this work is the parameter adjustment is not updated automatically depending on the variation of sensed values; the model is not adaptive.

Aderohunmu et al. (2013a,b)aim to increase the lifetime of a WSN by using naive, weighted moving average, ARIMA, and exponential smoothing forecasting methods. Our framework uses the forecasting methods employed by Aderohunmu et al. (2013a). However, its method selection depends on some factors which do not include the current statistical features, so it may be not achieved highest data reduction.

Moghadam and Keshmirpour (2011) aim to reduce the energy consumptions of wireless sensors nodes by using hybrid ARIMA and neural network model. The sensor node uses ARIMA model to evaluate the forecasted value. If the difference between the actual value and the forecasted value exceeds a certain threshold, the sensor uses the neural network for enhancing the forecasting outputs. In the case of the difference becomes less than the threshold, the sensor sends a small message size (a beacon signal) to the destination node to notify that ARIMA model should use the output of the neural network. On the other hand, if the difference exceeds the threshold, the ARIMA model is re-trained, and actual reading and the model parameters are transmitted to the destination node. The main drawbacks of this model are the large size of the packet is transmitted when the model failure in the forecasting. The packet is not only containing the actual readings as our framework but also containing the ARIMA model parameters (p is the order of the autoregressive model, q is the order of the moving average model, and d is the order of differencing). Also, the time and energy which are consumed in the retraining and forecasting process.

Papageorgiou et al. (2015) aim to solve I/O bottleneck problem of IoT data by applying real-time data reduction at the network edge. They presented the NECtar agent, which is able to select suitable reduction method for different IoT data types. Our framework has been compared with the two works cited above as shown in Section 5.

3. Proposed framework architecture

This section introduces an adaptive framework that targets the reduction of the amount of data transferred between SMs and either the cloud (in direct AMI) or aggregators (in indirect AMI). The framework consists of two layers: the smart meter layer and the data aggregation layer as shown in Fig. 1.

- The smart meter layer: includes SMs that are capable of measuring and recording energy consumption in real-time.
- The data aggregation layer: includes either the cloud (in direct AMI) or aggregators (in indirect AMI) that are responsible for collecting SMs data.

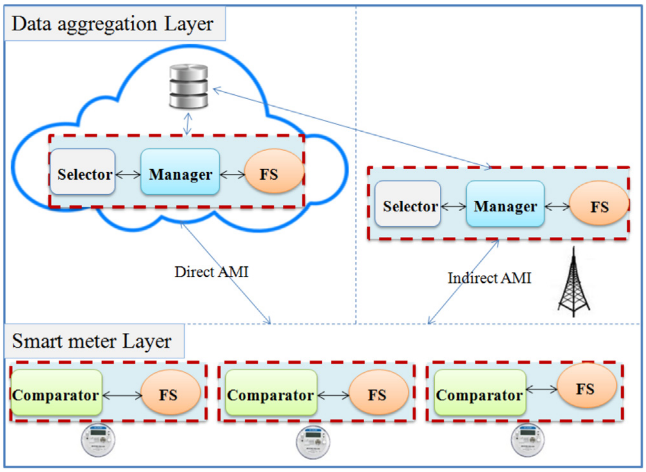

Fig. 1. Adaptive data reduction framework.

The proposed AMI architecture is composed of five components; two components (comparator and forecasting service FS) in the SM layer and three components (forecasting service, selector and manager) in the data aggregation layer (highlighted by the dashed rectangle in Fig. 1).

The main features of our framework are selecting an appropriate forecasting method for each consumer (time series) and adaptively change the currently used forecasting method based on the times series features. These targets are achieved by, first, the time is divided into equal periods (e.g. 10, 20, or 30 days) called switching periods. At the beginning of switching period, the time series of the previous period is analyzed, and its statistical features are extracted. Second, training the historical data for estimating the most appropriate methods (N, MA or ES). Based on the training step, there are two cases can be occurred, (i) only one method is always the best for the time series, so it is selected for the incoming forecasting, or (ii) more than one methods are the best for the time series, therefore building a decision tree is mandatory for generating method selection rules. Third, depending on generated rules from a decision tree and extracted features of the previous switching period, the appropriate forecasting method is selected for the current period.

In order to guarantee the same forecasting values between both FSs in SM and data aggregation layers. For each SM, both FSs have the same list of elements named forecasting list Z. It should be noted that this list is mixed of actual and forecasted values. It is initialized by SM readings at the beginning of switching period. Then, if the forecasted value is close to actual value then both FSs record the forecasted value otherwise they record the actual value at the top of the forecasting list.

3.1. Comparator

The comparator receives the actual consumption $x(n)$ at an instant n from SM. Then it asks the FS to evaluate forecasted value $y(n)$ using the same forecasting method of the data aggregation layer and forecasting list Z. Afterwards, it evaluates the error $e(n)$ as $|y(n) - x(n)|$. Finally, it compares $e(n)$ with error threshold e_max defined by the utility center. If $e(n) \leqslant e\_max$, the comparator does not send the actual reading $x(n)$ to the data aggregation layer but discards and replaces it with the forecasted value $y(n)$ in the forecasting list $Z(n) = \{y(n), z(n-1), z(n-2), \ldots z(n-m)\}$. On the other hand, if the $e(n) > e\_max$, the comparator sends the actual reading $x(n)$ to the data aggregation layer and discards the forecasted value $y(n)$ in the forecasting list as $Z(n) = \{x(n), z(n-1), z(n-2), \ldots, z(n-m)\}$.

$$z(n) = \begin{cases} y(n), & e(n) \leqslant e\_max \\ x(n), & e(n) > e\_max \end{cases} \tag{1}$$

3.2. Forecasting service

The forecasting service FS (FS_A in the SM layer and FS_B in the data aggregation layer) applies the forecasted method using the forecasting list Z. For this purpose, three different lightweight short-term time-series forecasting methods have been investigated (Aderohunmu et al., 2013a).

- Naïve (N): forecasted value $y(n)$ of the energy consumption is set to be the value of the last observation.

$$y(n) = z(n-1) \tag{2}$$

- Moving Average (MA): forecasted value $y(n)$ of the energy consumption is set to be the value of the average of the last k observations. Where k is an integer number greater than one.

$$y(n) = \frac{\sum_{j=1}^{k} z(n-j)}{k} \tag{3}$$

- Exponential Smoothing (ES): forecasted value $y(n)$ of the energy consumption is set to be a combination of $z(n-1)$ and $y(n-1)$ values.Where λ represents the smoothing parameters, in the interval $0 \leqslant \lambda \leqslant 1$.

$$y(n) = \lambda z(n-1) + (1-\lambda)y(n-1) \tag{4}$$

3.3. Selector

The selector component is responsible for two tasks (i) building a decision tree to generate rules for methods selection, as will be shown later (ii) analyzing the consumption data of each SM and extracting the following statistical time series features (Wang et al., 2009):

1. Average.
2. Standard deviation (Std).
3. Trend pattern: exists when there is a long-term change (increase or decrease) in consumption data.
4. Seasonality pattern: exists when there is a pattern of consumption data repeating itself over a fixed and known period.
5. Autocorrelation: represents the degree of similarity between a given consumption data and a lagged version of itself over successive time intervals.
6. Skewness: measures the degree of asymmetry of the consumption data.
7. Kurtosis: measures whether the consumption data are peaked or flat.
8. Terasvirta's NN test: measures the amount of non-linearity of the consumption data.
9. Hurst exponent: measures the self-similarity within consumption data.
10. Lyapunov Exponent: measures the amount of chaos in consumption data.

Afterwards, the selector chooses an appropriate method among (N, MA, or ES) according to statistical features by using the selected decision tree algorithm (e.g C4.5, CART, or Hoeffding tree according to their accuracy).

3.4. Manager

The manager component receives actual consumption $x(n)$ at an instant n from the comparator and saves it in a database and forecasting list Z. In case of missing data, it invokes the FS_B to approximate actual consumption value $y(n)$ according to the selected forecasted method and saves it in both database and Z.

The manager also notifies the selector to choose an appropriate forecasting method for each SM at the beginning of the switching period. Then it compares the data reduction percentage (DRP) of the current used forecasting method $DRP1$ and the selected method by the selector $DRP2$ of the previous switching period. If the difference between $(DRP2 - DRP1)$ is greater than a switching threshold α (e.g. 1,2, and 3%), then the manager sends the name of forecasting method to both FSs in SM and data aggregation layers. Otherwise, the manager ignores the selected method and does not send anything to FSs.

4. Framework workflow

The workflow of the proposed framework can be divided into two phases: method selection and reduction. The method selection phase is responsible for analyzing the consumption data and selecting an appropriate forecasting method for each SM. The reduction phase is to minimize the amount of data transferred between SMs and either the cloud (in direct AMI) or aggregators (in indirect AMI). These two phases are explained in the following subsections.

4.1. Method selection phase

Method selection phase includes two modes; decision tree construction mode and execution mode:

In the first mode, the selector builds a decision tree to select an appropriate forecasting method for each SM. The decision tree is built as follows:

1. For each SM, the energy consumption (kWh) measured in fixed intervals are collected.
2. The data reduction percentage is evaluated by various forecasting methods (N, MA, and ES) with different parameters given the statistical features for each switching period with length L.
3. The winning method for each switching period are determined according to the highest data reduction percentage and switching threshold α.
4. The classification dataset is built as the feature vector is formed by the statistical features and labeled by its winning method.
5. The decision tree algorithm is applied to generate rules and extract the relation between the winning method and statistical feature. The algorithm (C4.5, CART, or Hoeffding tree) with highest classification accuracy (percent correct, IR-precision, IR-recall, F-measure, and ROC area) will be applied.

Notably, in order to enhance the classification accuracy, we apply ensemble learning algorithms. If the winning methods (class labels) are unbalanced; this is a sample from one class is far less than the sample of other classes (Longadge and Dongre, 2013), this unbalance should be treated by ensemble learning algorithms. In this paper, AdaBoost, Bagging, MetaCost, and SMOTEBoost have been used which are considered the most commonly used ensemble learning algorithms (Galar et al., 2012).

In execution mode, at beginning of a switching period, the manager component notifies the selector component to update the forecasting method for each SM. The selector analyzes the previous switching period data, extracts the important statistical features, and chooses an appropriate forecasting method depending on the collected knowledge (decision tree). The selected method is sent to the manager. Then it compares between the reduction of selected and current used method of the previous switching period, if the difference between two methods is greater than a switching threshold α, it broadcasts FMethod and MeterID to both FS replicas. Fig. 2 views the sequence diagram of execution mode.

4.2. Reduction phase

The comparator receives the actual consumption $x(n)$ from SM, it invokes the FS_A to approximate actual consumption value $y(n)$. Afterwards, the FS_A evaluates the forecasted value $y(n)$ and send it back to the comparator. The comparator compares between actual consumption value $x(n)$ and the forecasted value $y(n)$ from FS_A; if $|x(n) - y(n)| \leqslant e\_max$, the $x(n)$ will not be sent to the manager and a top of forecasting list $z(n)$ is set to be equal $y(n)$, else the $x(n)$ will be transmitted and $z(n)$ is set to be equal $x(n)$.

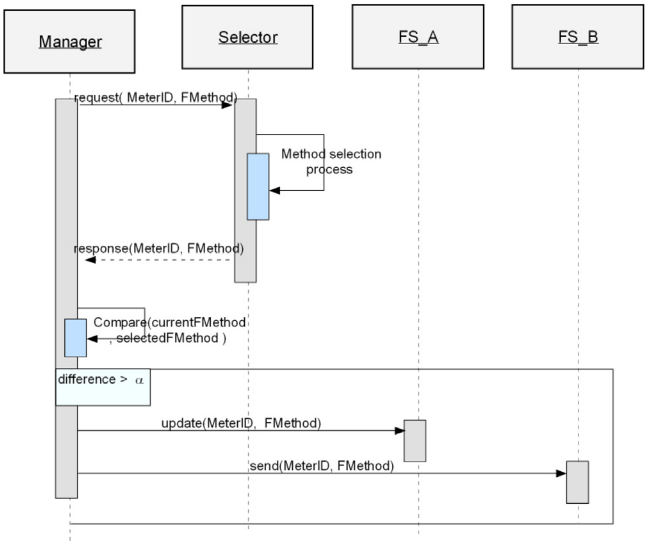

Fig. 2. The sequence diagram of execution mode.

Finally, manager component receives consumption data $x(n)$ from the comparator and saves it in both the database and at the top of forecasting list. In the case of missing data, the manager invokes FS_B to approximate actual consumption $y(n)$ value and saves it in both database and at top of forecasting list. Fig. 3 views the sequence diagram of the reduction phase.

5. Experimental results

The following subsections will introduce the used dataset in our experiments, the performance measures, and the detailed experimental scenarios.

5.1. Dataset

The SM data used in this paper were collected by (Creative Commons Attribution 3.0 Australia) (Sample household electricity time of use data, 2016). The SM measure the energy consumption every 30 min. We extract from the data set, the energy consumption of 10 consumers for one year. The total number of records collected for each consumer is 17520 records. Each record for each consumer includes the time and energy consumption measured by a SM as shown in Table 1.

The consumption data for each consumer represent a time series which indexed from $t1$ to $t10$. Each series can be characterized by a statistical feature as shown in Table 2. As mentioned above, each series divided into switching periods with length L. Fig. 4 views an example of half-hourly energy consumption of one switching period with $L = 30$ days.

5.2. Performance measures

For the sake of this work, we use more aspects to verify and assess our results as data reduction percentage, accuracy (Papageorgiou et al., 2015) and decision tree accuracy (percent correct, IR-precision, IR-recall, F-Measure, and ROC Area) (Bal and Sharma, 2016scikit-learn, 2017).

- Data Reduction Percentage (DRP) is obtained by the percentage of the data not transferred (NT) over total data sensed (S) by SM.

$$DRP = \frac{NT}{S}.100\% \tag{5}$$

- Data Reduction Accuracy (DRA): the accuracy between the sensed data by SM (X) and new time series (mixed of forecasted and actual data) (Z) is defined based on jaccard coefficient. Where m is the length of the time series.

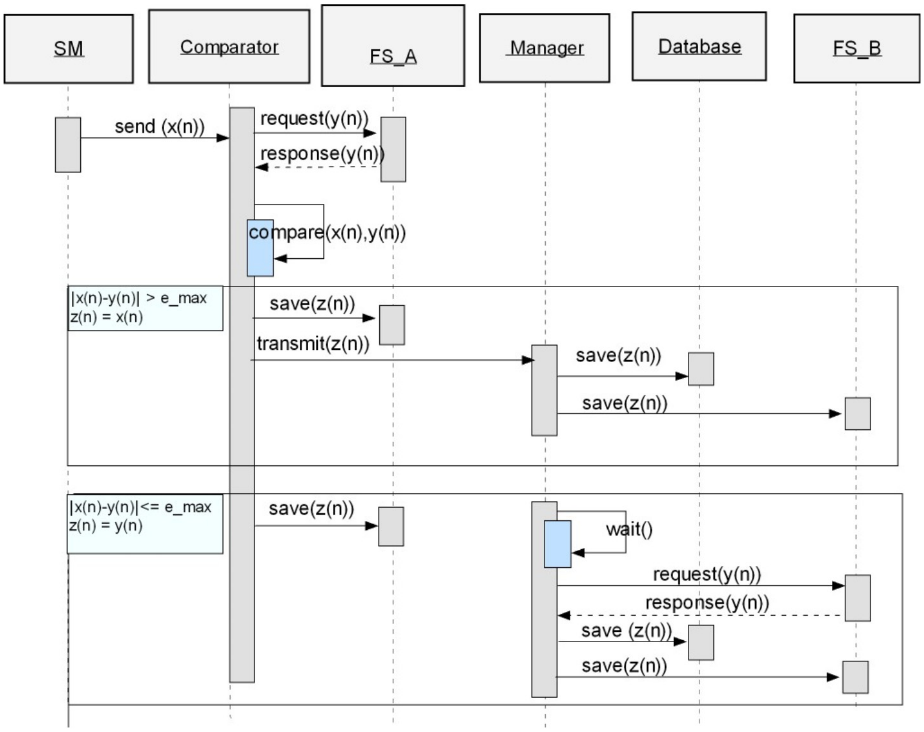

Fig. 3. The sequence diagram of reduction phase.

Table 1
An example of the SM energy consumption dataset.

Time	Energy consumption (kWh)
22/01/2013 9:29	0.321
22/01/2013 9:59	4.172
22/01/2013 10:29	3.946
22/01/2013 10:59	3.902

$$DRA = \frac{\sum_{i=1}^{m} \min(X_i, Z_i)}{\sum_{i=1}^{m} \max(X_i, Z_i)} .100\% \qquad (6)$$

- Percent correct: it is the percentage of correct classification.

- IR-precision: it is known as information retrieval precision. It is correctly classified positives over total forecasted as positives. High precision indicates that the decision tree algorithm returned substantially more relevant results than irrelevant.

- IR-recall: it is known as information retrieval recall. It is correctly classified positives over total positives. High recall indicates that the decision tree algorithm returned most of the relevant results.

- F-Measure: a weighted average of the precision and recall, where an F-Measure reaches its best value at one and worst at zero.

- ROC Area: or simply ROC curve is a graphical plot that clarifies the performance of classification algorithms. It is constructed by plotting the fraction of true positives out of the positives (TPR = true positive rate) vs. the fraction of false positives out of the negatives (FPR = false positive rate), at different threshold settings.

5.3. Experimental results

Five experiments were carried out using R and Weka tools. R version 3.2.5 (Team, 2014) is used to evaluate the statistical features, DRP, and DRA. Weka 3.7 tool (Hall et al., 2009) is used to build decision trees. These experiments are divided into three categories:

- First category (Experiment 1) examines decision trees construction and accuracy.
- Second category (Experiments 2, 3, & 4) examines the relationship between the framework parameters (error threshold *e_max*, switching period length L, and switching threshold α) and (DRP and DRA). In these experiments, we select only 3 of the 10 time series from Table 2 , namely $t4$, $t8$, and $t10$. In order to study the effect of the statistical features on (DRP and DRA).
- Third category (Experiment 5) apply our framework on the same data set used in Papageorgiou et al. (2015) and Moghadam and Keshmirpour (2011)to compare our results with them.

Notably, from Experiment 1 to 4, we divided each time series ($t1$ to $t10$) into 2 series, first one includes 8 months ($8 * 30$ days) of consumption data for building a decision tree, and the other one includes 4 months ($4 * 30$ days) for evaluating the DRP and DRA.

5.3.1. Experiment 1

In this experiment, a decision tree construction is built using the following parameters: *e_max* $= 0.01$ kWh, switching period length $L = 30$ days and switching threshold $\alpha = 2\%$. The winning methods depending on these parameters and used data set are (N wins 43 times, MA(2) wins 32 times and ES(.1) wins 5 times); a total number of records 80 (10 consumers $*$ number of switching period 8). Note that, the number of switching periods evaluated by dividing the length of training time series (8 months = 240 days) over the length of switching period L (30 days).

Fifteen classifiers named (C4.5, CART, Hoeffding tree, Adaboost C4.5, Adaboost CART, Adaboost Hoeffding tree, Bagging C4.5, Bagging CART, Bagging Hoeffding tree, MetaCost C4.5, MetaCost CART, MetaCost Hoeffding tree, SMOTEBoost C4.5, SMOTEBoost CART, and SMOTEBoost Hoeffding tree) are applied to build the decision tree. Then, we apply non-parametric tests on multiple groups, all vs all, as (Friedman, Friedman Aligned Ranks, and Quade) (Rodrguez-Fdez et al., 2015) for comparing the results shown in Table 3. Based on the ranking results, the best algorithms are (BaggingC4.5, Adaboost CART, Bagging CART, and SMOTEBoost CART), and the worst algorithms are (MetaCost Hoeffding, Hoeffding, and SMOTEBoost Hoeffding). This means that the criteria of the CART which selects the best splitting feature (entropy) (Bal and Sharma, 2016), is the best for the used dataset. On the other hand, Hoeffiding tree performs worst because Hoeffiding tree is more efficient with high volume, rate, and unbounded data set (Fang, 2015). However, the data set of this experiment is bounded and has low volume.

5.3.2. Experiment 2

In this experiment, we examine the impact of the error threshold *e_max* on the DRP and DRA. Other framework parameters are fixed, the length of the switching period L is 30 days, and the switching threshold α is 2%. In Fig. 5, on the left-hand side, DRP is plotted against *e_max*. DRP reaches 78% based on *e_max* and the time series features. In general, when *e_max* increases the DRP does too. SM will accept more forecasted values as actual value in the case of large *e_max* which increases DRP. On the other hand, on the right-hand side, DRA is plotted against *e_max*. As we see, DRP reaches up to 99.9% with lowest *e_max* (0.01 kWh). In general, the lower *e_max* the higher DRA. This is logical, since if the *e_max* is low the error between the actual readings and new series (mixed of actual and forecasted readings) becomes low which increases DRA.

Table 2
The statistical features for ten consumers (time series).

	Trend	Seasonal	Autocor-relation	Non-linearity	Skewness	Kurtosis	Hurst	Lyapunov (Chaos)	Average	Std
t1	0.0004	0	0.2576	1	0.9944	1	0.7642	0.9757	0.0824	0.0891
t2	0.0006	0	0.3210	1	0.7276	1	0.9317	0.9834	0.1777	0.1297
t3	0.0006	0	0.7090	0.9992	0.9679	1	0.9999	0.9940	0.1777	0.2219
t4	0.0025	0	0.5504	1	0.9839	1	0.9998	0.9963	0.1927	0.3856
t5	0.0995	0	0.7589	0.9775	0.4968	0.9472	0.9999	0.9945	0.6884	0.4313
t6	0.0184	0	0.5905	0.8037	0.7689	0.9996	0.9999	0.9967	0.6675	0.5074
t7	0.0037	0	0.0983	1	0.8030	0.9998	0.7388	0.9970	0.3802	0.5265
t8	0.4183	0.0018	0.9315	0.8979	0.8008	0.9980	1	0.5306	0.5037	0.5667
t9	0.0002	0	0.5577	1	0.6668	0.9913	0.9993	0.9975	0.5254	0.6060
t10	0.2566	0.5648	0.9224	0.9999	0.8829	1	1	0.5305	0.7069	1

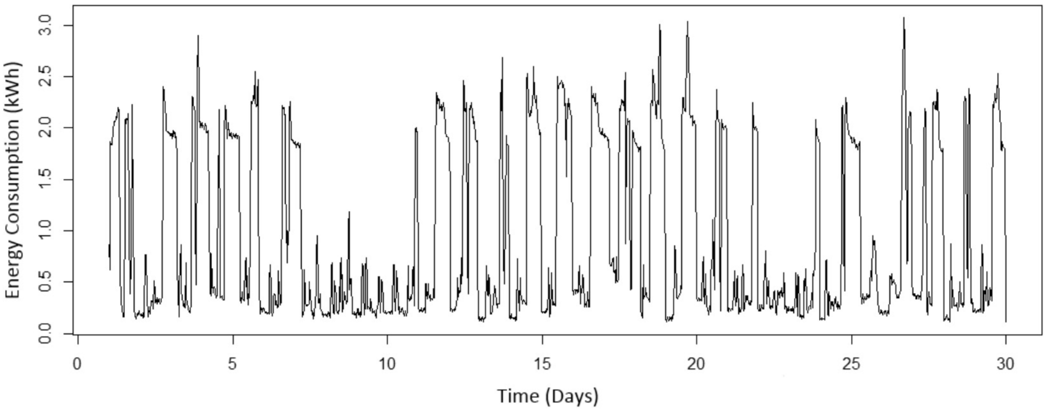

Fig. 4. An example of real incremental energy consumption readings over a period of 30 days (Sample household electricity time of use data, 2016). Each reading represents the consumption in the preceding half an hour.

Table 3
Fifteen decision tree algorithms in the terms of different comparison fields.

	Percent correct	IR-recall	IR-precision	F-Measure	ROC Area
C4.5	67.63	0.70	0.72	0.69	0.71
CART	68.38	0.74	0.72	0.70	0.71
Hoeffding tree	59.25	0.51	0.68	0.55	0.70
Adaboost C4.5	74.88	0.79	0.77	0.76	0.79
Adaboost CART	78.75	0.83	0.81	0.80	0.81
Adaboost Hoeffding tree	62.38	0.66	0.66	0.63	0.68
Bagging C4.5	78.50	0.84	0.80	0.81	0.84
Bagging CART	76.63	0.84	0.78	0.79	0.85
Bagging Hoeffding tree	61.75	0.56	0.69	0.59	0.72
MetaCost C4.5	69.50	0.71	0.75	0.71	0.72
MetaCost CART	72.13	0.75	0.76	0.74	0.73
MetaCost Hoeffding tree	58.88	0.44	0.68	0.51	0.65
SMOTEBoost C4.5	77	0.80	0.80	0.78	0.79
SMOTEBoost CART	76.75	0.83	0.79	0.79	0.81
SMOTEBoost Hoeffding tree	48	0.74	0.54	0.57	0.61

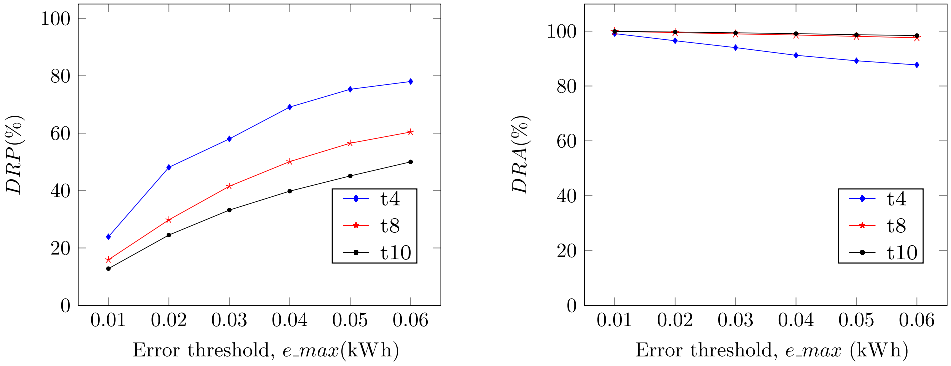

Fig. 5. *DRP* and *DRA* vs. *e_max* for three time series.

5.3.3. Experiment 3

In this experiment, we examine the impact of the switching period length L on DRP and DRA. Other framework parameters are fixed, the *e_max* is 0.02 kWh, and the switching threshold α is 2%. In Fig. 6, DRP and DRA are plotted against L. DRP can go up to 48% with DRA of 96.5%. DRP for $t10$ does not change. However, DRP for $t4$ and $t8$ have a very slightly increase when L increases. The increase in data reduction occurs given the fact that long periods lead to better feature extraction which improves the method selection.

5.3.4. Experiment 4

In this experiment, we examine the impact of the switching threshold α on DRP and DRA. Other framework parameters are fixed, *e_max* is 0.02 kWh, and the length L of the switching period is 30 days. In Fig. 7, DRP and DRA are plotted against α respectively. DRP reaches up to 48% with DRA of 96.5%. Also, the impact of the switching threshold α has been found to be minimal. This occurs as the difference between the currently used method and the selected method by selector is small or as the customer behavior in 4 months did not change, hence, switching method is not required.

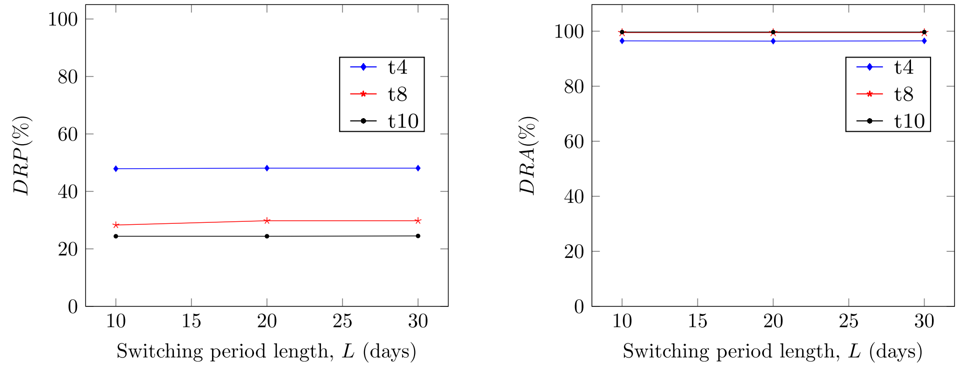

Fig. 6. *DRP* and *DRA* vs. *L* for three time series.

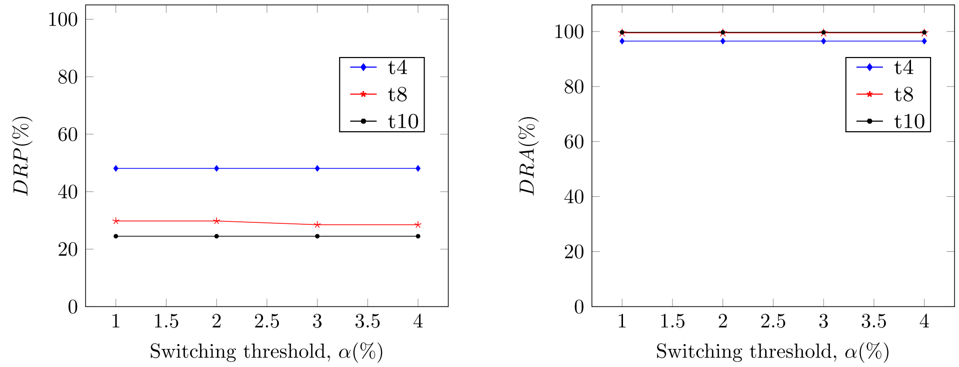

Fig. 7. *DRP* and *DRA* vs. α for three time series.

It is observed that *t*4 with the highest DRP (48%) and lowest DRA (96.5%) has the lowest value of mean and standard deviation. The *t*10 with the lowest DRP (24.5%) and highest accuracy (99.7%) has the highest value of mean and standard deviation. This logically leads to conclude that the framework with forecasting methods (N, MA, and ES) is more efficient when the consumption data values are within limited range. However, it is less efficient when the consumption data values have large standard deviation.

5.3.5. Experiment 5

In order to assess our framework, we compare our results with Papageorgiou et al. (2015) and Moghadam and Keshmirpour (2011), aiming at the same target of real-time data reduction for uni-variate time series. We apply our framework to the dataset Bache and UCI (2014) and Madden (2018) used in the work cited above respectively.

- First time series (TS1) is from a highway ramp loop sensor that counts the number of cars passing. The loop sensor reading is every 5 min, over a period of 175 days. The total number of readings is 50400 readings. We divided this time series into 2 series, the first series is 30 days for training, and the second series is 145 days in order to evaluate DRP and DRA. The following parameters are fixed in this experiment: $e\_max = 6$ cars, $L = 5$ days and $\alpha = 2\%$.
- Second-time series (TS2) represents the energy consumption of an individual household. The SM readings are every minute, over a period of 61 days. The total number of readings is $(61 * 24 * 60) = 87840$ readings. The attributes selected by Papageorgiou et al. 2015 from this dataset were not specified. Hence, we apply our framework to all these attributes. We divided each attribute (time series) into 2 series, the first series is 30 days for training, and the second series is 31 days for evaluating DRP and DRA. The following parameters are fixed in this experiment: L = 1 day, $\alpha = 2\%$, and e_max = 20 kW for active and reactive power, 0.07 kV for voltage, 1 A for intensity and 0.001 kWh for submetering 1, 2 and 3. Table 4 and 5 show that our framework provides better DRP and DRA than (Papageorgiou et al., 2015). The proposed framework leads to DRP up to 98.6% with DRA of 96.7%.
- Third-time series (TS3) represents temperature, humidity and voltage of the Intel Berkeley Research Lab (IBRL) for 52 sensors. The sensor reading is every 31 secs, over 1 day (2004-03-04). The total number of readings extracted for this comparison is $(52 * 650) = 33800$ readings. We divided the extracted data into 2 series, the first series is $(50 * 52) = 2600$ readings for training, and the second series is $(600 * 52) = 31200$ for evaluating the number of transmitted packets and size of transmitted packets. The following parameters are fixed in this experiment: $L = 50$ readings, $\alpha = 2\%$, and $e\_max = 0.3$ for temperature and humidity, and 0.03 for voltage. Assuming that, sensor reading size is 4 bytes, each ARIMA model parameters is 2 bytes, and beacon signal is 1 byte, we can estimate the size of packets transmitted.

As shown in Table 6, the results of Moghadam et al. are better than the result of the proposed framework in the number of transmitted packets as ARIMA and neural network enhance the forecasting accuracy. So, the number of transmitted packets is decreased. However, our framework is better than Moghadam et al. in the size of transmitted packets per sensor. Because of its packet is not only containing the actual sensor readings as our framework but also containing the ARIMA model parameters. Moreover, the time and energy which are consumed in the retraining and forecasting process.

Table 4
Comparison between NECtar data handlers and proposed framework for TS1(Highway Ramp Loop Sensor).

	Data Handler	Readings	DRP	DRA
Papageorgiou et al. 2015	NECtar-IPH-Clone	41851	68.8%	78.7 %
	NECtar-IPH-Twin	41851	65.6%	85.7 %
	NECtar-IPH-Avg	41851	65.6%	81.8 %
Our framework	Forecasting Methods	41760	73.25%	90.12%

Table 5
Comparison between NECtar data handlers and proposed framework for TS2(Household Smart Meter).

	Data Handler	Readings	DRP	DRA
Papageorgiou et al. 2015	NECtar-IPH-Clone	44642	66.3%	93.8 %
	NECtar-IPH-Twin	44642	66%	92.9 %
	NECtar-IPH-Avg	44642	63.3%	80.1 %
Our framework on				
Global active power	Forecasting Methods	44640	66.2%	99.4%
Global reactive power		44640	85.6%	90.6 %
Voltage		44640	66.2%	98.6%
Global intensity		44640	70.9%	99.7%
Sub-metering 1		44640	95.8%	96.8%
Sub-metering 2		44640	84.3%	88.4%
Sub-metering 3		44640	98.6%	96.7%

Table 6
Comparison between Hybrid ARIMA and neural network model and proposed framework for TS3(IBRL).

		Temperature	Humidity	Voltage
Moghadam and Keshmirpour (2011)	Number of model updates	852	1654	322
	Number of packet transmitted	868	1687	348
	The size of packets transmitted	8536	16573	3246
Our framework	Number of packet transmitted	1749	2463	734
	The size of packets transmitted	6996	9852	2972

6. Discussions and future work

Recently, many countries plan to replace traditional power grids with smart grids as it enhances the reliability, economics, efficiency, and sustainability of electricity services (Baek et al., 2015). One of the main challenges of this renovation is how to transfer a huge amount of data from SM to the data aggregation layer in real-time and high rate (e.g. every 15 min), it requires high communication bandwidth and stable internet connection. Transferring SM data rapidly and reliably is mandatory for real-time applications e.g. (real-time pricing) (Cook et al., 2012).

Data reduction techniques are widely used for overcoming the previous challenges as it can save the energy and communication bandwidth. Mainly, data reduction techniques can be classified into three types: data forecasting, data compression, and in-network processing. Data compression and in-network processing techniques are not considered in this paper because they require to accumulate a number of SM readings before the compression or processing is carried out. So, they do not support real-time data reduction (Papageorgiou et al., 2015).

On the other hand, in data forecasting techniques, a forecasting service (FS) is replicated at both the SM and data aggregation layers. The SM compares the actual energy consumption and the forecasted value by (FS). If the difference between the forecasted value and actual value is greater than a predefined threshold, SM sends the latter to the data aggregation layer. Instead, no data is transmitted, and the data aggregation layer will depend on the forecasted consumption generated by its FS. In other words, both FSs are acting as semi-active replication (Mohamed, 2016). So, we can observe that the real-time reduction is applicable by these techniques.

In data forecasting techniques, the forecasting is not only occurred in the SM layer, but also in the data aggregation layer. To achieve an accurate reduction, the input of both forecasting services in SM and data aggregation layers must be the same. To further illustrate how the reduction is achieved, Table 7 views the details of an example 20 elements extracted from the (Creative Commons Attribution 3.0 Australia) (Sample household electricity time of use data, 2016). By applying the Naïve (N) forecasting method and using $e\_max = 0.011$ kWh. DRP = 80% and DRA= 90.8% are reached.

Extending the input of the forecasting services (e.g. weather condition, shares of electrical and electronical companies, and the ratio of young population), of course, would increase the forecasting accuracy and in accordance the reduction percentage. However, this extension requires a bigger memory and better battery capabilities on the SM side in order to process the new data. Furthermore, it needs tight synchronization between the SM and data aggregation layers that are typically costly and complex.

In our framework, uni-variate time series forecasting methods (N, MA, and ES) are used as they (i) require neither external data nor extended analysis to make accurate forecasting (independent methods), (ii) require less memory; the maximum length of FS list is five which are required for MA(5), (iii) provide real-time computation which is mandatory as the data reduction occurs in real-time, and (iv) capture data nonlinearities efficiently.

Using other methods as ARIMA or machine learning techniques (e.g. artificial neural networks (ANNs)) could lead to better forecasting accuracy. However, in case of these model are failed in the forecasting, these models are required to retrained, and transmits the sensor reading and new parameters to the destination node which cause time, energy, and bandwidth consuming.

Table 7
Example illustrate how the reduction is achieved.

#	Actual Reading	Forecasting (N)	Absolute Different	Transmit if(Diff >0.011 kWh)	SM list	Data aggregation list
1	0.037	0	0	1	0.037	0.037
2	0.048	0.037	0.011	0	0.037	0.037
3	0.04	0.037	0.003	0	0.037	0.037
4	0.044	0.037	0.007	0	0.037	0.037
5	0.044	0.037	0.007	0	0.037	0.037
6	0.04	0.037	0.003	0	0.037	0.037
7	0.047	0.037	0.01	0	0.037	0.037
8	0.036	0.037	0.001	0	0.037	0.037
9	0.094	0.037	0.057	1	0.094	0.094
10	0.074	0.094	0.02	1	0.074	0.074
11	0.055	0.074	0.019	1	0.055	0.055
12	0.046	0.055	0.009	0	0.055	0.055
13	0.045	0.055	0.01	0	0.055	0.055
14	0.061	0.055	0.006	0	0.055	0.055
15	0.051	0.055	0.004	0	0.055	0.055
16	0.06	0.055	0.005	0	0.055	0.055
17	0.056	0.055	0.001	0	0.055	0.055
18	0.062	0.055	0.007	0	0.055	0.055
19	0.045	0.055	0.01	0	0.055	0.055
20	0.051	0.055	0.004	0	0.055	0.055

In the future work, we propose to develop a framework aiming to real-time data reduction for multi-variate time series. We plan to analysis multi-variate time series to find the relationships between distinct variables and then we can forecast one more variables from others.

7. Conclusions

In this paper, an adaptive framework is proposed to minimize the amount of data transferred between SMs and either the cloud (in direct AMI) or aggregators (in indirect AMI) in real-time. This is achieved by (i) building a decision tree to find the relation between the forecasting methods and statistical features of consumption data, (ii) analyzing the time series of consumption data to extract statistical features, and (iii) selecting an appropriate forecasting method based on the data features. The forecasting process is done on different replicas placed on both SM and (cloud or aggregators). When the actual value of energy consumption is close to the forecasted value, the SM does not need to send the actual value. Experimental results show that the proposed framework leads to a data reduction percentage (DRP) up to 98% with an accuracy (DRA) of 96% according to consumer behavior (statistical features of energy consumption data) and the operational parameters of the framework. The proposed framework has outperformed state of art methods (Papageorgiou et al., 2015) in DRP and DRA.

References

Aderohunmu, F.A., Paci, G., Brunelli, D., Deng, J.D., Benini, L., 2013a. Prolonging the lifetime of wireless sensor networks using light-weight forecasting algorithms, 2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing. IEEE, pp. 461–466.

Aderohunmu, F.A., Paci, G., Brunelli, D., Deng, J.D., Benini, L., Purvis, M., 2013b. An application-specific forecasting algorithm for extending wsn lifetime, 2013 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS). IEEE, pp. 374–381.

Alahakoon, D., Yu, X., 2015. Smart electricity meter data intelligence for future energy systems: a survey. IEEE Trans. Industr. Inf. 12 (1), 425–436.

Anastasi, G., Conti, M., Di Francesco, M., Passarella, A., 2009. Energy conservation in wireless sensor networks: a survey. Ad hoc networks 7 (3), 537–568.

Arbi, I.B., Derbel, F., Strakosch, F., 2017. Forecasting methods to reduce energy consumption in WSN, Instrumentation and Measurement Technology Conference (I2MTC), 2017 IEEE International. IEEE, pp. 1–6.

Bache, K., Lichman, M., UCI machine learning repository, 2014. [online] Available at: http://archive.ics.uci.edu/ml [Accessed 13 June 2017].

Baek, J., Vu, Q.H., Liu, J.K., Huang, X., Xiang, Y., 2015. A secure cloud computing based framework for big data information management of smart grid. IEEE Trans. Cloud Comput. 3 (2), 233–244.

Bal, R., Sharma, S., 2016. Review on Meta Classification Algorithms using WEKA. Int. J. Comput. Trends Technol. (IJCTT) 35 (1), 38–47.

Carvalho, C., Leal, L., Lemos, M., Holanda, R., 2014. Avoiding Data Traffic on Smart Grid Communication System. In Proceedings of the 1st Int. Electron. Conf. Sens. Appl., 2014; Sciforum Electronic Conference Series, 2014, 1. doi: https://doi.org/10.3390/ecsa-1-g007.

Chauhan, H., Chauhan, A., 2013. Implementation of decision tree algorithm c4.5. Int. J. Sci. Res. Publ. 3 (10), 1–3.

Chen, F., Deng, P., Wan, J., Zhang, D., Vasilakos, A.V., Rong, X., 2015. Data mining for the internet of things: literature review and challenges. Int. J. Distrib. Sens. Netw. 2015, 12.

Cook, B., Gazzano, J., Gunay, Z., Hiller, L., Mahajan, S., Taskan, A., Vilogorac, S., 2012. The smart meter and a smarter consumer: quantifying the benefits of smart meter implementation in the United States. Chem. Cent. J. 6 (S1), S5.

Cui, C., Wu, T., Hu, M., Weir, J.D., Li, X., 2016. Short-term building energy model recommendation system: a meta-learning approach. Appl. Energy 172, 251–263.

Dias, G.M., Bellalta, B., Oechsner, S., 2016. A survey about prediction-based data reduction in wireless sensor networks. ACM Comput. Surveys (CSUR) 49 (3), 58.

Fang, V., 2015. Hoeffding Tree for Streaming Classification. Available at:https://victorfang.wordpress.com/2015/09/23/hoeffding-tree-for-streaming-classification/ [Accessed 22 Dec 2017].

Foreman, J.C., Pacheco, F., 2016. Aggregation Architecture for Data Reduction and Privacy in Advanced Metering Infrastructure. arXiv preprint arXiv:1607.06377.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F., 2012. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(4), pp. 463–484.

Ghasempour, A., 2015. Optimized scalable decentralized hybrid advanced metering infrastructure for smart grid, 2015 IEEE International Conference on Smart Grid Communications (SmartGridComm). IEEE, pp. 223–228.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H., 2009. Weka 3: Data mining software in java. The University of Waikato. [online] Available at: http://www.cs.waikato.ac.nz/ml/weka/ [Accessed 13 June 2017].

Hulten, G., Spencer, L., Domingos, P., 2001. Mining time-changing data streams, Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 97–106.

Kabalci, Y., 2016. A survey on smart metering and smart grid communication. Renew. Sustain. Energy Rev. 57, 302–318.

Kang, J.J., Luan, T.H., Larkin, H., 2016. Enhancement of Sensor Data Transmission by Inference and Efficient Data Processing, International Conference on Applications and Techniques in Information Security. Springer, Singapore, pp. 81–92.

Longadge, R., Dongre, S., 2013. Class imbalance problem in data mining review. arXiv preprint arXiv:1305.1707.

Madden, S., Intel Berkeley research lab data. USA: Intel Corporation, http://db.lcs.mit.edu/labdata/labdata.html. [accessed 16 Feb. 2018]

Moghadam, R.A., Keshmirpour, M., 2011, International Conference on Informatics Engineering and Information Science. Springer, Berlin, Heidelberg, pp. 35–48.

Mohamed, M.F., 2016. Service replication taxonomy in distributed environments. SOCA 10 (3), 317–336.

Papageorgiou, A., Cheng, B., Kovacs, E., 2015. Real-time data reduction at the network edge of Internet-of-Things systems, 2015 11th International Conference on Network and Service Management (CNSM). IEEE, pp. 284–291.

Rasanen, T., Kolehmainen, M., 2009. Feature-based clustering for electricity use time series data, International Conference on Adaptive and Natural Computing Algorithms. Springer, Berlin Heidelberg, pp. 401–412.

Rodrguez-Fdez, I., Canosa, A., Mucientes, M., Bugarn, A., 2015. STAC: a web platform for the comparison of algorithms using statistical tests, 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE, pp. 1–8.

Sample household electricity time of use data, Creative Commons Attribution 3.0 Australia.https://data.gov.au/dataset/sample-household-electricity-time-of-use-data/ [Accessed 1 Apr. 2016].

scikit-learn, Model evaluation: quantifying the quality of predictions. [online] Available at: http://scikit-learn.org/stable/modules/modelevaluation.html. [Accessed 13 June 2017]

Team, R.C., 2014. The R project for statistical computing. Available at: http://https://www.r-project.org/ [Accessed 13 June 2017].

Wang, X., Smith-Miles, K., Hyndman, R., 2009. Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. Neurocomputing 72 (10), 2581–2594.

Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Philip, S.Y., Zhou, Z.H., 2008. Top 10 algorithms in data mining. Knowledge Inf. Syst. 14 (1), 1–37.

Yan, Y., Su, W., 2016. A fog computing solution for advanced metering infrastructure, Transmission and Distribution Conference and Exposition (T & D), 2016 IEEE/PES. IEEE, pp. 1–4.

Zeinali, M., Thompson, J.S., 2016. Impact of compression and aggregation in wireless networks on smart meter data, 2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). IEEE, pp. 1–5.