Vietnam Journal of Computer Science
https://doi.org/10.1007/s40595-018-0119-7

REGULAR PAPER

Short-term load forecasting in smart meters with sliding window-based ARIMA algorithms

Dima Alberg[1] · Mark Last[2]

Received: 10 August 2017 / Accepted: 13 November 2017

Abstract
Forecasting of electricity consumption for residential and industrial customers is an important task providing intelligence to the smart grid. Accurate forecasting should allow a utility provider to plan the resources as well as to take control actions to balance the supply and the demand of electricity. This paper presents two non-seasonal and two seasonal sliding window-based ARIMA (auto regressive integrated moving average) algorithms. These algorithms are developed for short-term forecasting of hourly electricity load at the district meter level. The algorithms integrate non-seasonal and seasonal ARIMA models with the OLIN (online information network) methodology. To evaluate our approach, we use a real hourly consumption data stream recorded by six smart meters during a 16-month period.

Keywords Internet of things · Smart city · Smart grid · Short-term forecasting · Incremental learning · Online information network · Sliding window · ARIMA

1 Introduction

Smart grid is becoming an increasingly popular application of the Internet of things (IoT). The smart grid includes a variety of operational and energy components connected to the Internet such as smart switches, smart meters, and smart appliances. Smart meters are aimed at monitoring and controlling household energy consumption in real time [2]. They enable two-way communication between the utility company and the customer. Their sampling rate usually varies from 10 min to 1 h with a maximum latency of 24 h.

The massive amounts of measurement data collected by smart meters can be used for customers' load forecasting. However, power consumption patterns in both residential and non-residential buildings may change over time due to multiple reasons including variability of human behavior, changes in the number of people populating the buildings at various times, introduction of new electric appliances, etc. Hence, short-term load forecasting (STLF) algorithms should be responsive to these changes by quickly learning new consumption patterns and modifying the forecasting models accordingly. The problem of a gradual "drift" in the target concept is handled by incremental learning systems via forgetting outdated data and adapting to the most recent phenomena [3]. However, the traditional ARIMA (auto regressive integrated moving average) algorithms, which are commonly used for short-term load forecasting, are lacking such an incremental learning mechanism: they learn the parameters of a given ARIMA model only once using a fixed training set and then apply that model to all future incoming data.

In this work, we integrate non-seasonal and seasonal ARIMA modeling with the OLIN (on line information network) incremental learning methodology, which was previously developed by Last [7] for classification tasks in the presence of a concept drift. The proposed incremental ARIMA system is aimed at continuously processing an infinite stream of incoming data such as a series of load measurements at an hourly or daily resolution. It periodically rebuilds the predictive model using the sliding window approach. We implement two non-seasonal and two seasonal sliding window-based ARIMA algorithms and evaluate them

✉ Mark Last
mlast@bgu.ac.il

Dima Alberg
albergd@gmail.com

[1] Department of Industrial Engineering and Management, SCE-Shamoon College of Engineering, Bialik St., Beer-Sheva, Israel

[2] Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev, 84105 Beer-Sheva, Israel

on a real-world consumption data stream recorded by six smart meters during a 16-month period.

2 Related work

Penya et al. [9] present short-term load forecasting models for non-residential buildings. According to the authors, this special domain presents different characteristics: there is no consumption at night, or it is negligible, and anyway, there exists a notable gap between idle and activity times. Another critical aspect is that usually, there is scarce (if any) historical data on hourly load and the load profile is sure to vary and evolve over the time. The forecasting results presented for the consumption data from a university campus shows that autoregressive models, being computationally simple, accurate, fast, and not requiring any trial-and-error customization or external data (e.g., temperature), are sufficient for providing acceptable prediction accuracy up to six days ahead (MAPE, mean absolute percentage error, between 5 and 11%). Other evaluated models, including linear and nonlinear regression, a neural network, a support vector machine, and a Bayesian network, have provided higher values of MAPE.

The latest studies presented in most recent articles provide contradictory results. In Høverstad et al. [6], ARIMA methods achieve better results than artificial neural networks. On the other side, Veit et al. [10] claim that a neural network performs slightly better than the ARIMA methods. Gerwig [4] points out that comparing the results of the different papers is difficult as the evaluations do vary not only in various consumption data sets but also in the length of the time series forecasting horizon, granularity, and the choice of error measures.

Gerwig [5] evaluates five state-of-the-art approaches to short-term load forecasting on three publicly available data sets of power consumption in residential buildings. The following forecasting methods are chosen for evaluation: an autoregressive model (AR), k-nearest neighbor regression (KNN), decision trees (DT), random forest regression (RF), and kernel ridge regression (KRR). In addition, two simple benchmarks are used: a persistent forecast (PER), where the predicted values are equal to the last observation, and an averaging method (AVG), where the predicted values are the average of the training data for the specific time of day. Compared to other methods, the autoregressive model and the KNN method achieve the best results (MAPE of about 30% in a 24-h forecast for a single household).

This paper contributes to our conference paper Alberg and Last [1] in terms of a more detailed explanation of the proposed algorithms and presentation of new results.

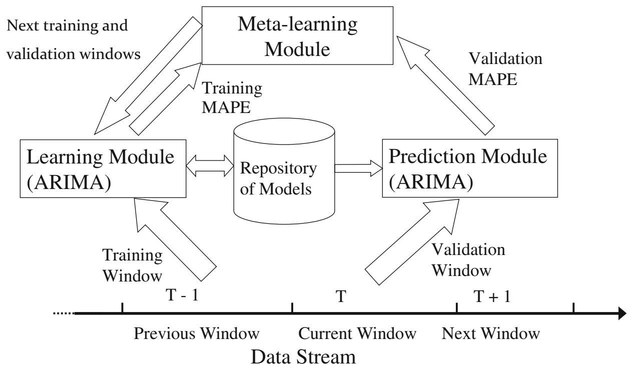

Fig. 1 Incremental OLIN learning with ARIMA

3 Proposed methodology

3.1 The incremental ARIMA paradigm

The proposed paradigm, called “Incremental ARIMA”, is presented in Fig. 1. Our incremental ARIMA system is composed of the following four components:

- *Learning module* it takes as input a sliding training window of a given size (in terms of the number of observations) and calculates the parameters for a given set of ARIMA models (seasonal or non-seasonal) as well as the training MAPE (mean absolute percentage error) of each induced model.
- *Repository of models* it serves for storing the ARIMA models induced from the latest training window.
- *Prediction module* it takes as input a sliding validation window of a given size (a “prediction horizon” such as the next 24 h) and calculates the validation MAPE (mean absolute percentage error) for each ARIMA model stored in the Repository of Models.
- *Meta-learning module* it takes as input the training and the validation MAPE of each [S]ARIMA model and chooses the most accurate model, which has the lowest value of validation MAPE. It also computes the start and the end points of the next training and validation windows, so that both the end point of a new training window and the start point of a new validation window are set to the end point of the previous validation window. To respond to a concept drift, a new [S]ARIMA model is induced from the latest training window every time the validation error of the current model exceeds its training error by a pre-defined threshold *Th*.

3.2 The incremental ARIMA algorithms

Seasonal autoregressive integrated moving average (SARIMA) models intend to describe the current behavior of variables in terms of linear relationships with their past values. These models are also called Box–Jenkins models following the Box and Jenkins (1984) pioneering work on time series forecasting techniques. A SARIMA model can be decomposed into four parts. First, it has an integrated (I) component (d), which represents the amount of differencing to be performed on the series to make it stationary. The second ARIMA component consists of an ARMA (autoregressive moving average) model for the series rendered stationary through differentiation. The third component is a seasonal component, and finally, the fourth component is the seasonality period parameter. The ARMA and SARMA components are further decomposed into the corresponding AR (autoregressive) and MA (moving average) non-seasonal and seasonal components, respectively. The AR and seasonal AR components capture the correlation between the current value of the time series and some of its past non-seasonal and seasonal adjusted values. For example, AR(1) means that the current observation is correlated with its immediate past value at time $t - 1$. The Moving Average MA and seasonal MA components represent the duration of the influence of a random non-seasonal and seasonal adjusted error. For example, MA(1) means that the error in the value of the series at time t is correlated with the shock at $t - 1$. The last thing to note is that most real-world time series are non-stationary, whereas ARIMA and SARIMA models usually refer to a stationary time series. Therefore, it is necessary to have a notational distinction between the original non-stationary time series and its stationary counterpart after differencing or logging.

We have used the paradigm described in the previous section to evaluate four incremental non-seasonal and seasonal (S)ARIMA algorithms: sliding window hourly ARIMA algorithm (SWH2A), sliding window hourly seasonal ARIMA algorithm (SWHSA), sliding window daily profile ARIMA algorithm (SWDP2A), and window daily profile seasonal ARIMA algorithm (SWDPSA). The SWH2A and SWHSA algorithms utilize hourly consumption records in the training window for calculating the parameters of hourly ARIMA and SARIMA models. Those models are applied recursively to each hour in the validation window for predicting the hourly consumption. On the other hand, the SWDP2A and SWDPSA utilize daily consumption records in the training window for calculating the parameters of daily ARIMA and SARIMA models. In addition, the hourly consumption records in the training window are used by SWDP2A and SWDPSA for calculating the 24-h average daily profile of hourly consumption. The daily ARIMA and SARIMA models are applied recursively to each day in the validation window for predicting the overall daily consumption and then combined with the mean 24-h profile for predicting the consumption during each hour.

The following flowchart gives an intuitive representation of the sliding window hourly ARIMA algorithm (SWH2A) and sliding window hourly seasonal ARIMA algorithm (SWHSA):

The pseudocode of the sliding window hourly ARIMA algorithm (SWH2A) and sliding window hourly seasonal ARIMA algorithm (SWHSA) is as follows:

```
Input
  • Training window size $W_{tr}$ (in hours)
  • Prediction (Validation) window size $W_{val}$ (in hours)
  • Starting time of the data stream $t_{start}$ (in hours)
  • ARIMA model type (p, d, q)
  • SARIMA model type (p, d, q) with seasonal component is (0,1,1)
  • Seasonality Period for SARIMA models is 24 hours
  • Th – concept drift threshold
Output
  • $MAPE_{Tr}$ – the training window MAPE
  • $MAPE_{Val}$ – the validation window MAPE
Algorithm
Initialize the start point of the training window $t_1 = t_{start}$
Compute the end point of the training window $t_2 = t_1 + W_{tr}$
$MAPE_{Val} = MAPE_{Tr}$
While new data arrives do:
        If ($MAPE_{Val} / MAPE_{Tr} > Th$)
              Induce hourly (S)ARIMA model from the hourly data in the training
              window [$t_1$; $t_2$]:
              (S)ARIMA-H = Model (hourlyData, p, d, q, $t_1$, $t_2$, (0, 1, 1), 24)
              Calculate $MAPE_{Tr}$ for the training window [$t_1$; $t_2$]:
              $MAPE_{Tr}$ = MAPE (hourlyData, (S)ARIMA-H, $t_1$, $t_2$)
        Compute the start point of the validation window $t_3 = t_2$
        Compute the end point of the validation window $t_4 = t_3 + W_{val}$
        Calculate $MAPE_{Val}$ for the validation window [$t_3$; $t_4$] :
        $MAPE_{Val}$ = MAPE (hourlyData, (S)ARIMA-H, $t_3$, $t_4$)
        Compute the start point of the training window $t_1 = t_4 - W_{tr}$
        Compute the end point of the training window $t_2 = t_4$
Loop
Return $MAPE_{Tr}$, $MAPE_{Val}$
```

The following flowchart represents the sliding window daily profile SWDP2A and SWDPSA algorithms as follows (Figs. 2, 3):

The pseudocode of the sliding window daily profile SWDP2A and SWDPSA algorithms is as follows:

```
Input
  • Training window size $W_{tr}$ (in days)
  • Prediction (Validation) window size $W_{val}$ (in days)
  • Starting time of the data stream $t_{start}$ (in days)
  • ARIMA model type (p, d, q)
  • SARIMA model type (p, d, q) with seasonal component is (0,1,1)
  • Seasonality Period for SARIMA models is 7 days
  • Th – concept drift threshold

Output
  • $MAPE_{Tr}$ – the training window MAPE
  • $MAPE_{Val}$ – the validation window MAPE
Algorithm
Initialize the start point of the training window $t_1 = t_{start}$
Compute the end point of the training window $t_2 = t_1 + W_{tr}$
$MAPE_{Val} = MAPE_{Tr}$
While new data arrives do:
    If ($MAPE_{Val} / MAPE_{Tr} > Th$)
            Induce daily (S)ARIMA model (S)ARIMA-D from the daily data in
            the training window [$t_1$; $t_2$]:
            (S)ARIMA-D = Model (dailyData, p, d, q, $t_1$, $t_2$)
            Calculate the 24-hour load profile hoursProfile from the hourly da-
    ta in the training window [$t_1$; $t_2$]
            Calculate $MAPE_{Tr}$ for the training window [$t_1$; $t_2$]:
            $MAPE_{Tr}$ = MAPE (hourlyData, hoursProfile, (S)ARIMA-D, $t_1$, $t_2$)
    Compute the start point of the validation window $t_3 = t_2$
    Compute the end point of the validation window $t_4 = t_3 + W_{val}$
    Calculate $MAPE_{Val}$ for the validation window [$t_3$; $t_4$] :
    $MAPE_{Val}$ = MAPE (hourlyData, hoursProfile, (S)ARIMA-D, $t_3$, $t_4$)
    Compute the start point of the training window $t_1 = t_4 - W_{tr}$
    Compute the end point of the training window $t_2 = t_4$
Loop
Return $MAPE_{Tr}$, $MAPE_{Val}$
```

The formula of the (S)ARIMA forecasting model depends on the following seven parameters [8]:

- p is the number of non-seasonal autoregressive terms.
- d is the number of non-seasonal differences.
- q is the number of lagged non-seasonal forecast errors in the prediction equation.
- ps is the number of seasonal autoregressive terms.
- ds is the number of seasonal differences.
- qs is the number of lagged seasonal forecast errors in the prediction equation.
- P is the seasonal periodicity.

For example, the forecasting equation of ARIMA (1, 1, 1) for the hourly/daily load during the hour/day t is:

$$\hat{Y}_t = \mu + Y_{t-1} + \phi(Y_{t-1} - Y_{t-2}) - \theta \cdot e_{t-1}, \tag{1}$$

where Y_{t-1} and Y_{t-2} stand for the actual load during the hours/days $t-1$ and $t-2$, respectively, e_{t-1} is the forecasting error for the hour/day $t-1$, and the coefficients μ, ϕ, and θ are estimated from the hourly/daily data in the training window using a model fitting technique. When the prediction horizon (validation window) exceeds one time unit (one hour or one day, respectively), the actual values of Y_{t-1} and Y_{t-2} in Eq. 1 above may be replaced with their forecasted values, so that (S)ARIMA model can be applied recursively to a series of observations.

3.3 The evaluated algorithms

The training/validation MAPE (mean absolute percentage error) of a given forecasting model is calculated on N consecutive load measurements as follows:

$$\text{MAPE} = \frac{\sum_{i=1}^{N} \left| \frac{\hat{Y}_i - Y_i}{Y_i} \right| \times 100}{N}, \tag{2}$$

where $\hat{Y}_i$ and Y_i stand for the predicted and the actual load, respectively.

The 24-h load profile is calculated from the hourly data in the training window by applying the following equation to each hour of the day:

$$P_h = \frac{\sum_{d=1}^{D} Y_{dh}}{D}, \tag{3}$$

where P_h is the average hourly load during the hour $h (h \in [1, 24])$, Y_{dh} is the actual load during the hour h of day d, and D is the number of days in the training window. The average daily load P_D can be found by summing up the values of P_h over all hours of the day:

$$P_D = \sum_{h=1}^{24} P_h. \tag{4}$$

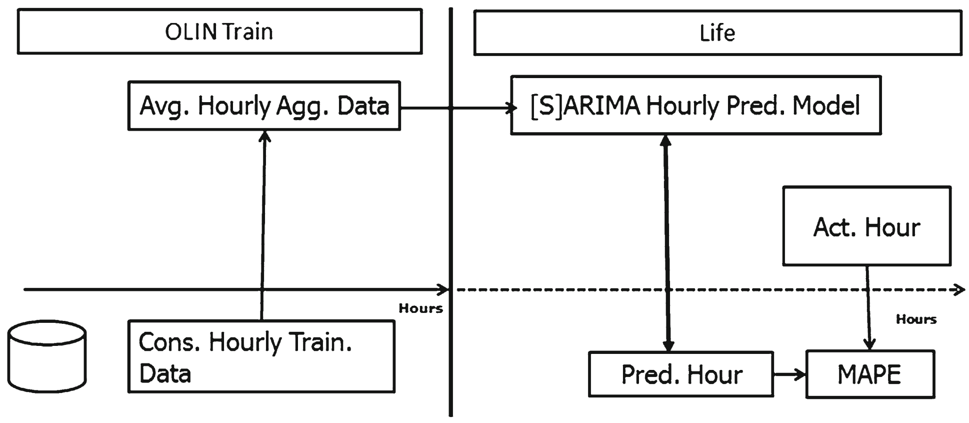

Fig. 2 SWH[2]SA flowchart algorithm representation

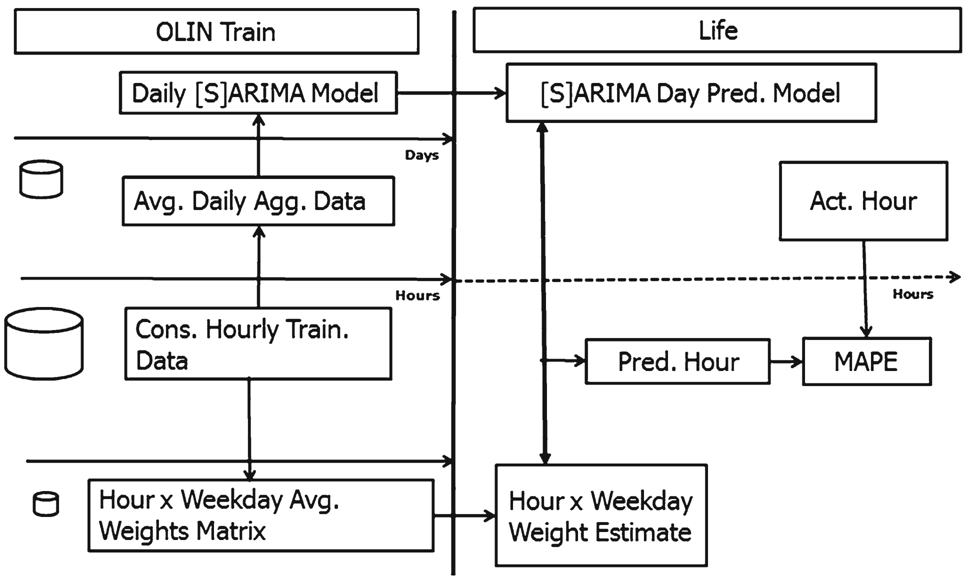

Fig. 3 SWDP[2]SA flowchart algorithm representation

The sliding window daily profile (S)ARIMA SWDP2A and SWDPSA algorithms use the following equation to predict the hourly load during the hour h of the day d:

$$\hat{Y}_{dh} = \frac{\hat{Y}_d}{P_D} \times P_h, \tag{5}$$

where $\hat{Y}_d$ is the predicted daily load for the day d calculated by a daily (S)ARIMA models, P_D is the average daily load calculated by Eq. 3 above, and P_h is the average hourly load for the hour h calculated by Eq. 2 above.

In our experiments, we have also evaluated two "naïve" models: the "naïve hourly" based on the hourly consumption during the previous day and "naïve daily profile" based on the daily consumption during the previous day and the average daily consumption profile during the training period. The "naïve hourly" model calculates the forecasted hourly load during the hour $h$of the day d by the following equation:

$$\hat{Y}_{dh} = Y_{d-1,h}, \tag{6}$$

where $Y_{d-1,h}$ is the actual load measured during the same hour h on the previous day $d-1$. The "naïve daily profile" model calculates the forecasted load during the day d by the following equation:

$$\hat{Y}_d = Y_{d-1}. \tag{7}$$

Then, it estimates the hourly load during each hour of the day d using Eq. 4 above.

4 Evaluation experiments

4.1 Design of experiments

Our evaluation experiments were based on the electricity hourly load data recorded by six Powercom (http://www.powercom.co.il) meters during a 16-month period between 01/12/2012 and 31/03/2014. The meters were installed in different districts of a major Israeli city. The total number of recorded hourly observations was 61,646. The experiments with four algorithms (SWH2A, SWHSA, SWDP2A, and SWDPSA) included nine non-seasonal hourly and nine non-seasonal daily ARIMA models,[1] nine seasonal hourly SARIMA models[2] with the period parameter of 24 h, nine seasonal daily SARIMA models (see footnote 2) with the period parameter of 7 days, and two baseline models: the "naïve hourly" model and the "naïve daily profile" model. The experiments with the SWH2A, SWHSA, and the "naïve hourly" models included three sizes of the training window (24, 48, and 96 days) and four sizes of the validation window (24, 48, 72, and 96 h). The experiments with the SWDP2A, SWDPSA, and the "naïve daily profile" models included three sizes of the training window (24, 48, and 96 days) and three sizes of the validation window (1, 2, and 3 days). The total number of models evaluated with each algorithm was $4 \times 38 \times 3 \times 3 = 540$.

We have explored the seasonality behavior of all meters by building the average MW (megawatt) consumption plots for monthly, daily, and hourly cycles of collected data.

The plots showed in Figs. 4, 5, and 6 exhibit seasonality patterns in monthly, daily, and hourly profiles, respectively.

[1] ARIMA models: (000, 001, 100, 101, 010, 011, 111, 221, 222).

[2] SARIMA models: (000, 001, 100, 101, 010, 011, 111, 221, 222) (0,1,1).

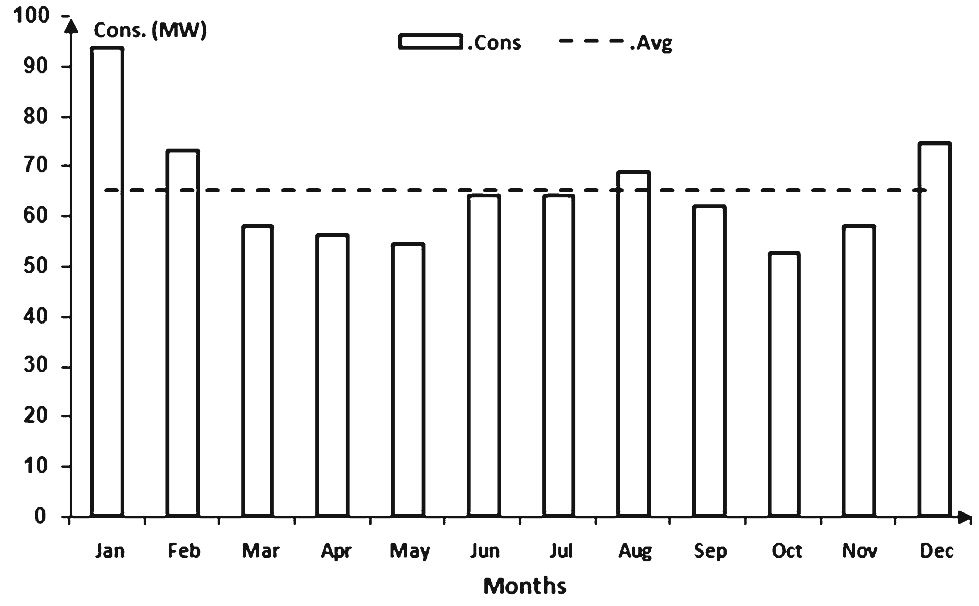

Fig. 4 Monthly consumption cycle for all meters

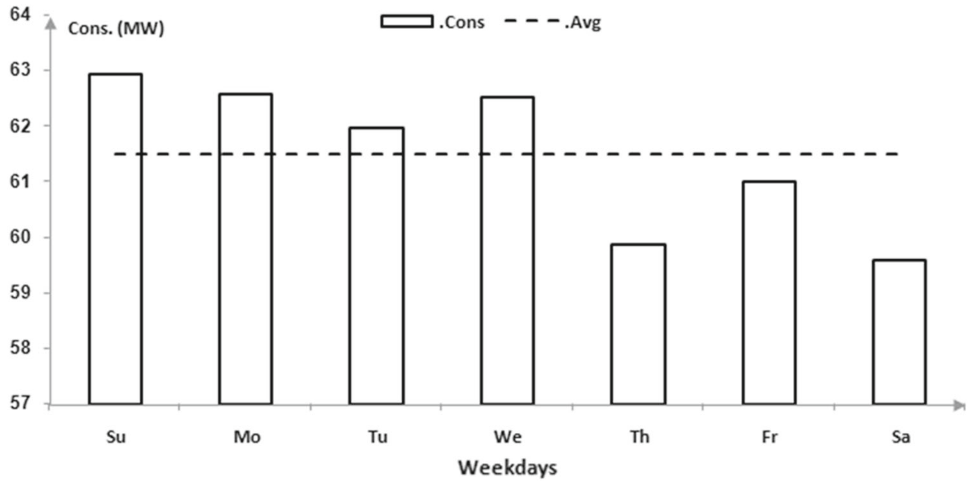

Fig. 5 Daily consumption cycle for all meters

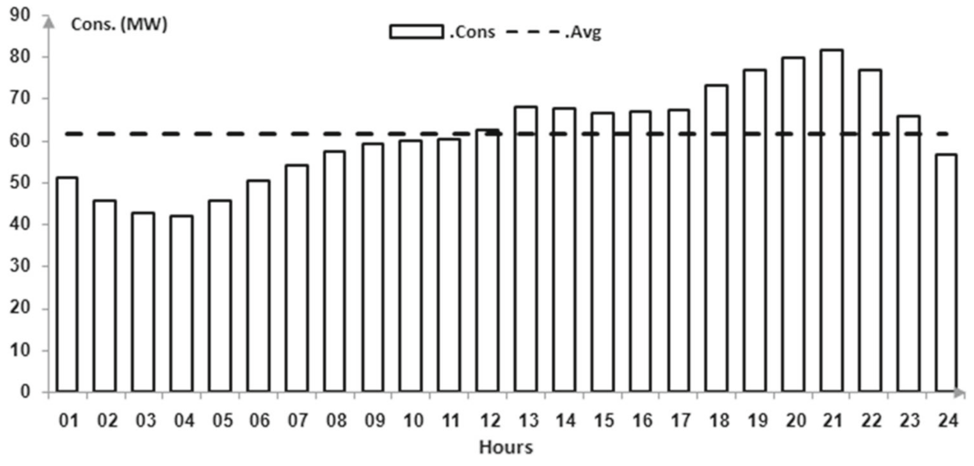

Fig. 6 Hourly consumption cycle for all meters

Figure 2 demonstrates a strong seasonality pattern expressed by high energy consumption in winter months versus other months. In addition, the months of June, July, and August represent relatively high and stable energy consumption. This fact coincides with the high temperatures in Israel when many residents are using air condition in their homes. Figure 5 demonstrates a weak daily data seasonality characterized by electricity load decrease in the last 3 days of the week (Thursday, Friday, and Saturday). This result is obvious, because Friday and Saturday are official weekend days in Israel and most organizations are closed, and consequently, they consume less electricity on these days. In contrast, Fig. 6 demonstrates a strong hourly seasonality pattern expressed by the electricity load increasing from 05:00 to 11:00 and then again between 17:00 and 21:00, particularly during the workdays (Sunday–Thursday). The first pattern may be related to the start of a business day at many workplaces, whereas the second one may be explained by many people returning home from work and starting to use the electrical appliances at their homes, while at the same time, many companies are starting their afternoon shifts. It is also noteworthy that after 21:00, the load starts decreasing as people are going to sleep and stop using most of their appliances.

4.2 Results

Table 1 compares the average performance of four (S)ARIMA and the "naïve" models across all meters and training/validation window sizes. The sliding window hourly ARIMA (SWH2A) models performed significantly worse than the other models: their average validation MAPE values are about two times higher than the "naïve hourly" MAPE (11.804%). Out of the sliding window daily profile ARIMA models (SWDP2A and SWDPSA), the best result (MAPE = 10.05%) is obtained with the non-seasonal SWDP2A ARIMA (101) model. Similarly, out of the sliding window hourly profile ARIMA models (SWH2A and SWHSA), the best result (MAPE = 9.19%) is obtained with the seasonal SARIMA (001) model. Figure 7 shows that the SWDP2A algorithm has a more stable MAPE performance in terms of standard deviation than the SWHSA algorithm.

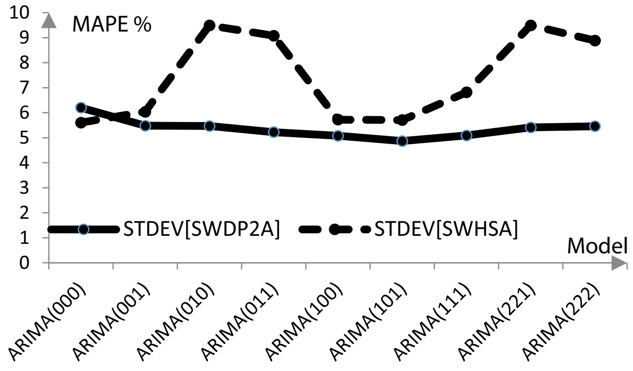

Fig. 7 Algorithm stability comparison in terms of standard deviation

Table 2 compares the average validation MAPE for each meter across various models and training/validation window sizes. In terms of the average MAPE, the daily profile model SWDP2A strongly outperforms the non-seasonal SWH2A hourly model (10.409 vs. 21.482%), slightly outperforms seasonal SWDPSA hourly model (10.409 vs. 11.451%), and has a similar performance to the SWHSA seasonal hourly model according to the paired sample *t* test at the 99% significance level.

Table 3 also shows the results of one-way ANOVA testing for statistical significance of the difference between the meters. The conclusion of one-way ANOVA is that the difference is not significant, implying that we can safely refer to consolidated results of all 6 m.

Table 4 compares the average validation MAPE for different sizes of the training window across various models, meters, and validation window sizes. It shows that, in general, increasing the training window size improves the forecasting performance (reduces MAPE), which indicates a relatively

Table 1 Comparison of ARIMA models in terms of Avg. MAPE

Model	SWDP2A	SWDPSA	SWH2A	SWHSA	Average
[000]	11.592	11.457	22.826	9.689	16.454
[001]	10.909	11.038	21.870	9.191	15.284
[010]	10.329	16.708	23.631	11.600	14.527
[011]	10.261	11.444	23.715	11.221	14.280
[100]	10.363	10.241	21.640	9.246	14.643
[101]	10.050	10.218	21.790	9.238	14.704
[111]	10.155	15.045	23.864	9.543	13.910
[221]	10.393	15.153	22.795	11.652	13.887
[222]	10.358	11.331	20.999	10.432	13.219
Naïve daily	10.779				
Naïve hourly			11.804		

Table 2 Comparison of meters (h)

Meter	SWDP2A	SWDPSA	SWH2A	SWHSA	Average
2478	9.627	11.331	20.083	9.740	13.215
4364	7.748	7.613	21.743	8.478	12.891
4429	14.618	15.675	21.231	15.293	16.902
4470	7.877	7.680	16.584	9.139	11.170
4740	11.042	11.643	30.980	11.559	17.260
5521	10.823	11.020	20.217	9.585	13.944
Average	10.409	11.451	21.482	10.622	14.230

Table 3 ANOVA comparison of meters

Source of Variation	SS	*df*	MS	*F*	*P* value	*F* crit
Between groups	134.41	5	26.88	0.73	0.61	2.77
Within groups	658.94	18	36.61			
Total	793.34	23				

Table 4 Comparison of training window sizes (h)

Train. window	SWDP2A	SWDPSA	SWH2A	SWHSA	Average
576	10.729	17.232	22.238	11.642	15.035
1152	10.325	15.136	21.533	11.011	14.573
2304	10.170	11.163	20.688	9.777	13.296
Average	10.409	11.451	21.482	10.622	14.230

Table 5 Comparison of validation window sizes (h)

Val. window	SWDP2A	SWDPSA	SWH2A	SWHSA	Average
24	10.009	11.211	20.533	9.532	13.052
48	10.443	11.515	21.334	10.324	13.665
72	10.787	11.642	21.957	11.213	14.177
96	9.047	11.331	22.103	11.428	18.036
Average	10.409	11.451	21.482	10.622	14.230

stable behavior of most meters during the period of at least 96 days (about three months). However, there was 1 m (4429) with the best training window size (providing the lowest value of MAPE) of 48 days and another meter (5521) with the best training window size of 576 h (24 days) only. Apparently, these 2 m were exposed to a faster concept drift than the other four ones.

Table 5 compares the average validation MAPE for different sizes of the validation window across various models, meters, and training window sizes. It shows that on average, extending the prediction horizon reduces the performance of the forecasting models (increases MAPE). Apparently, the rate of MAPE increase is going down for the SWH2A algorithm between the window sizes of 72 and 96 h. These results confirm the common knowledge that it is more difficult to predict a more distant future.

Finally, Table 6 shows the best configuration of algorithm, model, and training window size across all 6 m for each size of the validation window. The conclusion is that the seasonal SWHSA algorithm works best for the 24- and 48-h

Table 6 Best configuration for each prediction horizon (h)

Val. window	Min. Avg. MAPE	Algorithm	Model	Train. window
24	9.532	SWHSA	SARIMA(1,0,1)	2304
48	10.324	SWHSA	SARIMA(0,0,1)	2304
72	10.787	SWDP2A	ARIMA(1,0,1)	2304
96	9.047	SWDP2A	ARIMA(0,1,1)	1152

validation window sizes. For the two larger windows (48 and 72 h), the non-seasonal SWDPA algorithm induces the most accurate forecasting models. The maximum training window size of 96 days (2304 h) is the best one for the first three configurations. In case of the 96-h validation window (the fourth configuration), the best size of the training window is 48 days (1152 h).

5 Discussion and conclusions

The main contribution of this paper is the introduction of sliding window-based forecasting algorithms (SWDP2A, SWDPSA, SWH2A, SWHSA, and SWHSA) for electricity load prediction in smart meters. These algorithms integrate non-seasonal and seasonal time series (S)ARIMA models with the OLIN (online information network) incremental learning methodology. The main difference between the presented algorithms concludes in seasonality adjustment and the model construction phase. The non-seasonal SWH2A and seasonal SWHSA algorithms utilize hourly consumption records in the training window, whereas non-seasonal SWDP2A and seasonal SWDPSA algorithms utilize aggregated daily consumptions and average daily profiles of hourly consumptions to obtain the parameters of induced (S)ARIMA models.

The experimental data set was recorded online by state-of-the-art smart metering technology and, after thorough preprocessing, was approved for use in the corresponding research experiments. The conducted experiments showed that the SWDP2A algorithm outperforms the SW2SA algorithm, performs similarly to the seasonal SWHSA algorithm, and has more stable MAPE performance in terms of standard deviation than the SWHSA algorithm. This remarkable finding indicates that the hourly prediction task does not require collecting massive hourly data in the training phase of model induction. It is sufficient to use daily consumption data and aggregated hourly coefficients of daily profiles for obtaining accurate hourly predictions of electricity load.

Acknowledgements This work was partially supported by the Israel Smart Grid (ISG) Consortium under the MAGNET Program, in the office of the Chief Scientist of the Ministry of Economics in Israel.

References

1. Alberg, D., Last, M.: Short-Term Load Forecasting in Smart Meters with Sliding Window-Based ARIMA Algorithms. *ACIIDS2017*, (pp. 299-307) (2017)
2. Depuru, S.S., Wang, L., Devabhaktuni, V.: Smart meters for power grid: Challenges, issues, advantages and status. Renewable and Sustainable Energy Reviews **15**(6), 2736–2742 (2011)
3. Gama, J.: Knowledge Discovery from Data Streams. CRC Press, Boca Raton (2010)
4. Gerwig, C.: Short Term Load Forecasting for Residential Buildings - an Extensive Literature Review, pp. 181–193. Intelligent Decision Technologies, (2015)
5. Gerwig, C.: Short Term Load Forecasting for Residential Buildings. In J. Hu, & e. al., *Smart Grid Inspired Future Technologies: First International Conference, SmartGIFT 2016, Liverpool, UK, May 19-20, 2016, Revised Selected Papers* (pp. 69-78). Springer International Publishing (2016)
6. Høverstad, B., Tidemann, A., Langseth, H.: Effects of Data Cleansing on Load Prediction Algorithms. *In IEEE Symposium Series on Computational Intelligence 2013.* IEEE (2013)
7. Last, M.: Online Classification of Nonstationary Data Streams, pp. 129–147. Intelligent Data Analysis, (2002)
8. Makridakis, S.G., Wheelwright, S.C., Hyndman, R.J.: Forecasting Methods and Applications. John Wiley & Sons, New York, NY (2008)
9. Penya, Y., Borges, C., Fernandez, I.: Short-term Load Forecasting in Non-residential Buildings. *AFRICON 2011*, 9(3), pp. 1-6. Livingstone (2011)
10. Veit, A., Goebel, C., Rohit, T., Doblander, C., Jacobsen, H.: Household Electricity Demand Forecasting: Benchmarking State-of-the-Art Methods. *5th International Conference on Future Energy System* (pp. 233-234). ACM (2014)