IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Load Forecasting Under Concept Drift: Online Ensemble Learning with Recurrent Neural Network and ARIMA

Rashpinder Kaur Jagait[1], Mohammad Navid Fekri[1] , Katarina Grolinger[1], Syed Mir [2]
[1]Department of Electrical and Computer Engineering, Western University, London, ON N6A 5B9, Canada
[2]London Hydro, London, ON, Canada N6A 4H6

Corresponding author: Katarina Grolinger (e-mail: kgrolinger@uwo.ca).

This work was supported by the Ontario Centres of Excellence under Grant OCI: 33066

ABSTRACT Rapid expansion of smart metering technologies has enabled large-scale collection of electricity consumption data and created the foundation for sensor-based load forecasting on individual buildings or even the household level. With continuously growing energy consumption, the importance of energy management including load forecasting is increasing in order to remedy the energy effect on the environment. Numerous machine learning techniques have been proposed for sensor-based load forecasting but most are offline approaches: the model is trained once and then used to infer future consumption. However, these approaches are not able to adapt to concept drift: for example, their accuracy will degrade when the building use changes or new equipment is installed. Thus, an approach capable of learning from new data as they arrive is needed. This paper proposes adaptive online ensemble learning with Recurrent Neural Network (RNN) and ARIMA for load forecasting under concept drift. The RNN part of the ensembles consists of Online Adaptive RNN as its underlying RNN learner has the ability to model temporal dependencies present in load data while its online nature enables continuous learning from arriving data. The adaptation to the concept drift is improved by adding Rolling ARIMA to the ensemble. The performance of the proposed approach has been examined on the four individual homes with different degrees of concept drift. The results show that the proposed ensemble achieves better accuracy than its constituent algorithms alone and, moreover, the analysis demonstrates the need to examine load forecasting approaches in respect to how they handle concept drift.

INDEX TERMS load forecasting, concept drift, energy forecasting, ensemble learning, recurrent neural network, online learning

Acronyms

ADWIN	Adaptive Windowing
ARIMA	Autoregressive Integrated Moving Average
BN-RNN	Batch Normalized RNN
CMD	Connect My Data
DDM	Drift Detection Method
DL	Deep Learning
DM	Diebold-Mariano test
EIA	Energy Information Administration
FCC	Fully Connected Cascade
GRU	Gated Recurrent Unit
IMAE	Incremental Mean Absolute Error
LSTM	Long Short Term Memory
MA	Moving Average
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
PAR	Passive Aggressive Regression
RNN	Recurrent Neural Network
S2S	Sequence to Sequence
SBCTL	Similarity-based Chained Transfer Learning
TESLA	Taylor Expanded Solar Analog Forecasting

I. INTRODUCTION

U.S. Energy Information Administration (EIA) estimates that energy consumption will grow by 50% between 2018 and 2050 [1]. This expansion is largely driven by continuously increasing economic activities [2]. Economic

growth at an annual rate of 3–4% is not only driving energy demand but also generating the corresponding CO2 emission and negatively affecting the environment [2]. Energy management systems play a major role in mitigating energy production side effects by monitoring, controlling, and optimizing energy consumption and generation..

Load forecasting has been attracting tremendous research and industry interest because of its essential role in energy management systems: it is a foundation for energy generation and distribution planning, operation of supply, energy budgeting and it is an inseparable part of smart grid developments [3]. The expansion of smart meters which measure and record energy consumption has enabled energy forecasting on the building and even household level. A large number of smart meters in operation, over 70 million in the USA and over 96 million in China in 2016 [4], created opportunities for new deeper insights into energy usage patterns and enabled large-scale load forecasting. However, traditional forecasting techniques need to be examined in respect to how they handle diverse energy consumption patterns present among individual energy consumers as well as changes in patterns over time.

In recent years, Deep Learning (DL) approaches have demonstrated great successes in load forecasting because of their strong generalization capabilities, the ability to model complex systems, and extract features from row data [5]. Nevertheless, there is still a gap between conventional offline deep learning methods and the learning required to obtain insights from continuously arriving smart meter data. Conventional machine learning (ML) requires that the entire dataset be available at the start of the learning [6]. Then, this entire data set is passed repeatedly through the model and the model's parameters are adjusted; one pass over data is referred to as *epoch*. However, with smart meters, all data are not available at the start of training as data are continuously arriving. To acquire knowledge from the new data, the conventional model needs to be re-trained using all historical data combined with new data. Of course, this is impractical and computationally intensive as each time the model is re-trained from scratch.

Moreover, the underlying distribution of smart meter data changes over time, producing what is referred to as *concept drift* [7]. For example, adding a new appliance or a change in home occupants behaviour patterns will result in different energy consumption profiles. Nonetheless, traditional machine learning techniques are static, and in presence of the concept drift, they exhibit weak and degrading performance [8].

Therefore, for energy forecasting with smart meter data, we need an ML approach capable of learning from new data as they arrive over time without the need to re-train the model or keep historical data [9]. *Online learning*, a machine learning paradigm that uses data streams for training and learns one or a few instances at a time, has been proposed to address this challenge. The *'online'* descriptor reflects the fact that this paradigm continuously maintains its model and modifies the model as needed. This learn-as-you-go approach alleviates the computational load and removes the need for all data to be present at once [6]. Online learning has been applied for load forecasting: it has achieved better accuracy than traditional offline models with significantly reduced computation time [9]. As noted by Fekri *el al.* [9], online learning, because of its online nature, should be better at handling the concept drift; however, further examination of forecasting under concept drift is needed [9].

Consequently, this paper proposes an online ARIMA-RNN ensemble, a load forecasting approach capable of learning from new drifting data as they arrive. The approach combines two forecasting techniques: Online Adaptive Recurrent Neural Network (RNN) is used because of its online nature and demonstrated high accuracy in load forecasting [9], whereas Rolling ARIMA contributes to the model performance in presence of the concept drift as it is able to adapt quickly to changes in the load. The two techniques are combined using aggregation techniques ranging from simple averaging to adaptive weighted functions. Results show that the proposed online ARIMA-RNN ensemble outperforms standalone Online Adaptive RNN and standalone ARIMA in terms of overall forecasting error. Additionally, the paper examines the behaviour of the proposed approach for houses with different degrees of concept drift and highlights the importance of investigating algorithms' behaviour in presence of concept drift and across diverse consumers.

The remainder of the paper is organized as follows: Section II presents the related works, Section III discusses the background, Section IV describes the proposed approach, and Section V explains the experiments and corresponding results. Finally, Section VI concludes the paper.

II. RELATED WORK

Machine learning has been used extensively for load forecasting [10], [11] with techniques from the RNN category dominating in recent years.

Memarzadeh *et al.* [12] proposed Long Short-Term Memory (LSTM) based forecasting algorithm that uses wavelet transform to handle the fluctuations in electricity load. Also, their model employs entropy and mutual information methods to eliminate the redundant features. The proposed approach was evaluated on aggregated load forecasting for a region, therefore, it was not exposed to high data variability as is the case with individual households.

Sehovac *et al.* [13] proposed Sequence to Sequence Recurrent Neural Network (S2S RNN) with attention mechanism for load forecasting. The S2S model employs two RNNs, an encoder and a decoder, to map the input sequence to the output sequence. The attention mechanism strengthens the connection between the encoder and the decoder to assist with processing long sequences. Although S2S with attention has shown great results, it is computationally much more expensive than LSTM or even S2S without attention.

Eskandari *et al.* [14] proposed a CNN-GRU-LSTM model for hourly load forecasting considering external factors such as weather, weekday/weekend, and holiday. First, a Con-

volutional Neural Network (CNN) is utilized to extract the load and temperature features: at this stage, the univariate data are converted to multidimensional features by applying two-dimensional convolutional kernels. Next, these multi-dimensional features are passed to the bidirectional Gated Recurrent Units (GRUs) and LSTM units to generate the load forecasts.

Tian *et al.* [15] introduced Similarity-based Chained Transfer Learning (SBCTL) approach for load forecasting with a large number of smart meters. The proposed approach trains the initial model for the first smart meter in a traditional manner. Then, model training for all other meters uses transfer learning to take advantage of existing, already trained models based on similarities of consumers' load profiles. This technique generates a personalized model for each smart meter. In the experiment with 456 meters, SBCTL achieved similar accuracy to traditional individual model training while significantly reducing training time.

Several other studies also employed RNNs for various load forecasting tasks. Bidirectional RNN combined with a deep belief network was proposed for short-term load forecasting [16], [17]. Li *et al.* [18] also used bidirectional RNN for short-term load forecasting, but they added attention mechanism and distributed representation of input variables. Shi *et al.* [19] employed RNN, specifically LSTM, together with load profile pooling for residential load forecasting. LSTM and GRU units were also proposed for distribution feeder long-term load forecasting [20].

Ensemble techniques have also been proposed: Wang *et al.* [21] presented an ensemble learning approach for short and medium-range load forecasting. Their ensemble integrates a clustering method with LSTM and a Fully Connected Cascade (FCC) network. A clustering algorithm first partitions the historical data to train multiple LSTM models, and then the FCC model is used to fuse the trained LSTMs. The ensemble increased the prediction accuracy in comparison to standalone LSTM.

Gungor *et al.* [22] took an advantage of the ensemble technique for individual household electricity consumption prediction. In their approach, AutoRegressive Integrated Moving Average (ARIMA), Holt-Winters, TESLA (Taylor Expanded Solar Analog Forecasting), LSTM, and Persistence prediction algorithms are combined using a feed forward neural network to construct an ensemble. To decrease the computational cost of training, they applied pruning by eliminating small-valued weights from the network.

The reviewed techniques [12]–[23] are based on RNN variants such as LSTMs and GRUs because of their ability to model time dependencies. Although they achieved excellent accuracy in load forecasting, they all belong to the category of offline learning: once the model is trained, to acquire knowledge from new data, it must be re-trained from scratch with all historical data. In load forecasting, data from smart meters are continuously arriving and new data may have different patterns. Repeatedly re-training the model is computationally intensive and often infeasible. In contrast, our approach learns from data as they arrive without requiring re-training. Model parameters, in both Adaptive RNN and ARIMA, are dynamically adapted to capture the temporal changes in the underlying data streams. This dynamic nature makes the proposed ARIMA-LSTM well suited for forecasting under concept drift.

For wind forecasting, adaptive incremental linear regression was proposed [24]: as new stream samples arrive, the model learns gradually and endlessly. When the concept drift is detected, the window of past observations used for learning is reduced. Because this approach is based on linear regression, it is not suited for highly non-linear load data.

For forecasting streaming time series data in the presence of anomalies and change points, Guo *et al.* [25] devised an adaptive gradient learning method for RNNs. The approach wights the gradients based on the local distribution properties of new data. Although the approach demonstrated excellent results, it only works for one-step ahead forecasting. Madireddy *et al.* [26] highlighted the importance of considering concept drift for job scheduling in production systems. They proposed a concept drift aware prediction model: the location of the concept drift is detected with an online Bayesian changepoint detection method and then the training data collected before the drift are transformed by transfer learning-inspired technique for the model re-training.

Works of Vexler *et al.* [27] and Liang *et al.* [28] presented online learning approaches for load forecasting based on LSTM. Vexler *et al.* [27] combined LSTM and online density estimation with Hoeffding trees whereas Liang *et al.* [28] presented an approach for the smart grid with the model located at the network edge. Both studies [27], [28] did not specifically consider concept drift.

Krannichfeldt *et al.* [29] proposed an online load forecasting approach based on a modified Passive Aggressive Regression (PAR) model. Their technique actually combines batch and online learning: batch models provide individual forecasts, and the online ensemble combines their prediction to achieve adaptability. The proposed approach is well suited for smooth and convex problems, while non-linearities and a high level of complexity, including concept drift, are present in energy consumption data.

Similar to our study, the reviewed online and incremental learning techniques [24]–[29] learn continuously from data streams. However, they either do no consider concept drift [24], [27]–[29] or are limited in respect to concept drift they consider [25], [26]. Madireddy *et al.* [26] considered only abrupt drift while Guo *et al.* [25] only consider a single real-world dataset representing Yahoo services, which is a very different application and does not directly apply to load forecasting. In contrast, our research considers concept drift without a limitation in respect to the type of drift. Moreover, we consider diverse real-world scenarios and examine the behavior of the proposed system on data streams with low and high concept drift presence.

It is important to note that some studies impose assumptions that do not apply to residential load forecasting.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

Sánchez-Medina *et al.* [24] proposed a linear model while the approach proposed by Krannichfeldt *et al.* [29] is well suited for smooth and convex problems. On the other hand, non-linearities and a high level of complexity, including concept drift, are present in energy consumption data and our study investigates the performance of the proposed approach on complex real-world data sets.

Finally, Fekri *et al.* [9] proposed a non-linear approach for load forecasting and considered concept drift. Their Online Adaptive RNN uses Batch Normalized RNN (BN-RNN) as the base learner and combines Bayesian optimization, performance monitoring, and buffering to tune the BN-RNN model on the fly. As their approach achieved higher accuracy than the offline LSTM and several other online algorithms, we use it as one of the algorithms in our ensemble model. However, Online Adaptive RNN does not respond rapidly to changes in data when concept drift occurs, and, therefore, in our approach, it is assisted by rolling ARIMA.

III. BACKGROUND

This section introduces RNNs, ARIMA, ensemble learning, and Diebold-Mariano test.

A. RECURRENT NEURAL NETWORKS

Recurrent Neural Networks (RNNs) are among the state of the art deep learning algorithms for learning from sequential data [30]. They consist of RNN cells which are, in addition to the connection to neighbouring layers, also connected with a recurrent connection to the same cell at the previous time step. These connections together with the cells' internal memory make RNNs suitable for modelling temporal dependencies. However, vanilla RN suffers from vanishing gradient problem: a long data sequence causes exponential reduction of gradients as they are backpropagated through time and, as result, the network forgets older information.

To overcome this problem, Long Short Term Memory (LSTM) network illustrated in Fig. 1 was designed. In LSTM, added input i, forget f, and output o gates allow for better control over the gradient flows and assist in maintaining memory for longer periods of time. At the time t, LSTM computation is given as follows:

$$i_t = \sigma(W_{xi}x_t + b_{xi} + W_{hi}h_{t-1} + b_{hi}) \tag{1a}$$
$$f_t = \sigma(W_{xf}x_t + b_{xf} + W_{hf}h_{t-1} + b_{hf}) \tag{1b}$$
$$o_t = (W_{xo}x_t + b_{xo} + W_{ho}h_{t-1} + b_{ho}) \tag{1c}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xg}x_t + b_{xg} \tag{1d}$$
$$+ W_{hg}h_{t-1} + b_{hg}) \tag{1e}$$
$$h_t = o_t \odot \tanh(c_t) \tag{1f}$$

Here, c is the cell state, h is the hidden state, σ is the sigmoid activation function, and $\odot$ represents elementwise multiplication. The W_x's and W_h's are the input-hidden and hidden-hidden weights, respectively, and b_x's and b_h's are the corresponding biases.

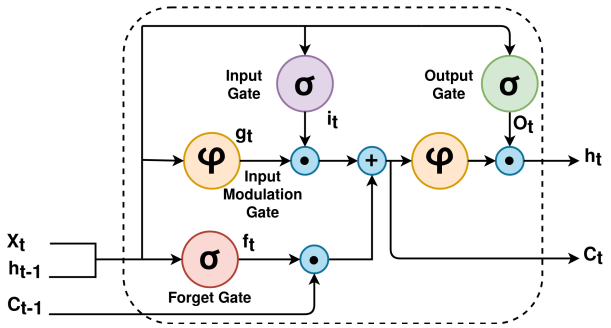

FIGURE 1. LSTM cell

This memory mechanism makes LSTM successful in load forecasting and energy prediction tasks. However, the LSTM is still an offline ML technique, and an adaptive online LSTM is required to modify itself quickly to reflect the new revealing patterns in data.

B. ARIMA

AutoRegressive Integrated Moving Average (ARIMA) models are fitted to past time series data to better understand data or to predict future values [31]. An ARIMA model is a combination of three parts: the Auto Regressive (AR) part represents the variable as a linear combination of its own lagged values, Moving Average (MA) denotes that the error is a linear combination of past errors, and Integrated (I) part refers to differencing applied to transform non-stationary time-series into stationary. ARIMA(p,d,q) denotes ARIMA with the order of autoregressive model p, the degree of differencing d, and order of the moving-average model q. For time series X_t where t is the time step, ARIMA(p,d,q) expresses the forecast value $\hat{X}_t$ as follows:

$$\hat{X}_t = \overbrace{\alpha_1 X'_{t-1} + \alpha_2 X'_{t-2} + \cdots + \alpha_p X'_{t-p}}^{\text{AR}} + \underbrace{e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}}_{\text{MA}} \tag{2}$$

where X' is the differentiated time series, α and θ are autoregressive and moving average coefficients, and e is the error term.

C. ENSEMBLE LEARNING

Ensemble learning is a machine learning paradigm that combines several base models to improve learning outcomes [6]. The main idea is that when the base models are strategically combined, the ensemble can achieve better outcomes than any constituent model.

The three main strategies for combining the base learners are: bagging, boosting, and stacking. The *bagging* [32] approaches train several independent base models and then aggregate their individual predictions by voting or by averaging to obtain the final prediction. *Boosting* [33] similarly uses several base models, however, unlike bagging which simply

aggregates the individual independent votes, boosting is an adaptive technique in which each base model depends on the previous ones. The final prediction is still obtained following a deterministic formula. Finally, *stacking* [34] differs from bagging and boosting in the way it combines the base models: the combining is carried out with a meta-model. In other words, the outputs of the base models are the inputs to another ML model which learns how to combine the base learners for better predictions.

D. DIEBOLD-MARIANO TEST

Although standard evaluation metrics such as Mean Absolute Error (MAE) and Mean Squared Error (MSE) are useful measurements for comparing models' results, they do not examine if the difference between the models is significant. Diebold-Mariano test [35], [36] can be employed to determine if forecasts are significantly different.

Suppose that we have two forecasts $f_1, \ldots, f_n$ and $g_1, \ldots, g_n$ for a time series $y_1, \ldots, y_n$, and we want to determine if the two forecasts are significantly different. Let $e_i = y_i - f_i$ and $r_i = y_i - g_i$ be the residuals (errors) for the two forecasts. The squared-error loss function is then defined as:

$$L_1 = \sum_{i=1}^{n} (e_i)^2 \tag{3}$$

$$L_2 = \sum_{i=1}^{n} (r_i)^2 \tag{4}$$

The null hypothesis is given as follows:

$$H_0 : E[L_1] = E[L_2] \tag{5}$$

where E is expectation value. Diebold-Mariano test is based on the loss differentials d_i:

$$d_i = e_i^2 - r_i^2 \tag{6}$$

Equivalently, the null hypothesis of equal predictive accuracy is shown as H_0: $E[d_i] = 0$. Then, the sample mean loss differential $\bar{d}$ is:

$$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_t = \frac{1}{n} [L_1 - L_2] \tag{7}$$

The autocovariance γ_k at lag k is defined as:

$$\gamma_k = \frac{1}{n} \sum_{i=1}^{n} (d_i - \bar{d})(d_{i-k} - \bar{d}) \tag{8}$$

Finally, the Diebold-Mariano statistic is shown as follow:

$$DM = \frac{\bar{d}}{\sqrt{[\gamma_0 + 2 \sum_{k=1}^{n^{\frac{1}{3}}+1} \gamma_k] n^{-1}}} \tag{9}$$

The null hypothesis is rejected at 5% confidence level every time the DM value is outside the range [-1.96 1.96].

IV. ONLINE ARIMA-RNN ENSEMBLE

This section proposes Online ARIMA-RNN Ensemble, an ensemble-based load forecasting approach that dynamically learns from evolving data streams and adapts to new patterns in the data. Online Adaptive RNN [9] is used as one of the base learners because of its ability to model time dependencies, online learning strategy, and demonstrated success in load forecasting. To improve the ensemble's ability to react to the concept drift, an ARIMA learner is added. Different ensembling strategies are explored including dynamic weighting based on past performance. The overview of the proposed Online ARIMA-RNN is shown in Fig. 2 with the details of the three main components, preprocessing, prediction models, and ensembler, described in the following subsections.

A. PREPROCESSING

Preprocessing is a traditional step in offline learning and involves techniques such as normalization and sample randomization. However, traditional data preprocessing techniques cannot always be applied to online learning as all data are not available at the start of learning. The preprocessing module in Online ARIMA-RNN Ensemble (Fig. 2) contains the preprocessing common for both, ARIMA and Online Adaptive RNN.

The data from smart meters are first transformed using the sliding window technique as shown in Fig. 3. The first window contains the first W smart meter readings and represents the first training sample. The next sample is obtained by sliding the window for one time step: it contains the readings from the time step 2 to $W + 1$. As in addition to the load readings, other attributes are used for forecasting such as the temperature and the day of the week, one sample is of dimension $W \times F$, where W is the number of time steps contained in the window, and F is the number of features. So far, this is the same as in offline learning, with the exception that the window samples are created gradually as data become available. Next, the *batch* is formed by grouping b consecutive samples created by the sliding window technique.

In offline learning, the windowing technique is used for creating the samples to feed into the model (often neural network). Here, this structure, together with batches, also assists in monitoring the model performance, and any significant change in performance triggers the model refinement and/or tuning. In Fig. 3, the windows $W1$ to $W4$ are before the concept drift, the windows $W5$ to $W9$ lie in the drifting period, and $W10$ to $W14$ belong to the new data patterns. Comparing the model performance across different windows will show degradation when the concept drift occurs indicating the need to adapt the model. For example, comparing between $W4$ and $W5$ or between $W9$ and $W10$ is expected to indicate the change in data and should trigger the model adaptation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

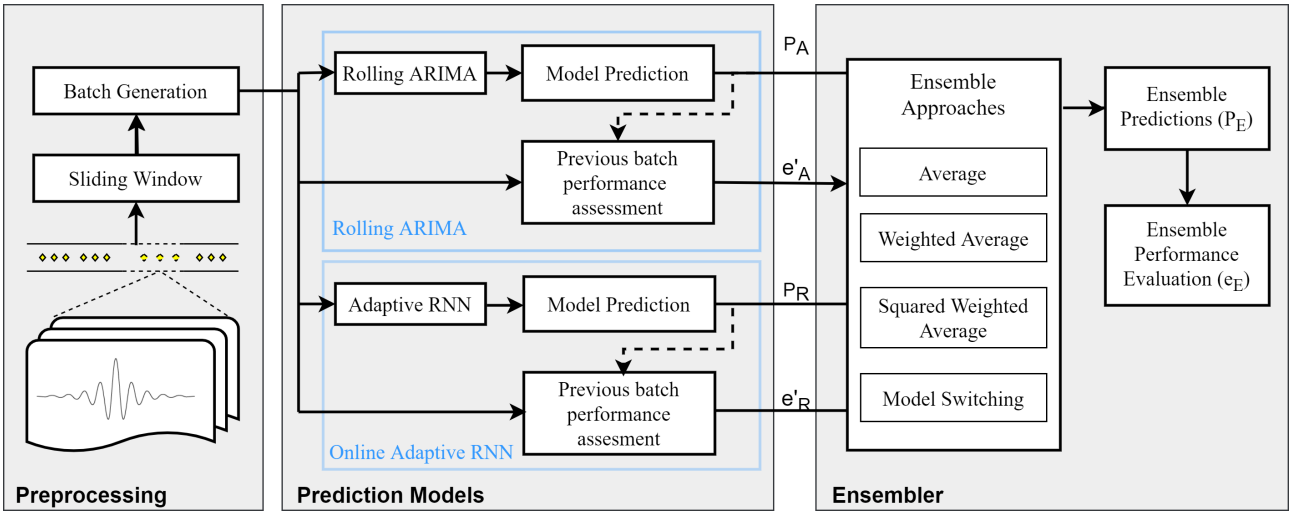

FIGURE 2. Online ARIMA-RNN Ensemble.

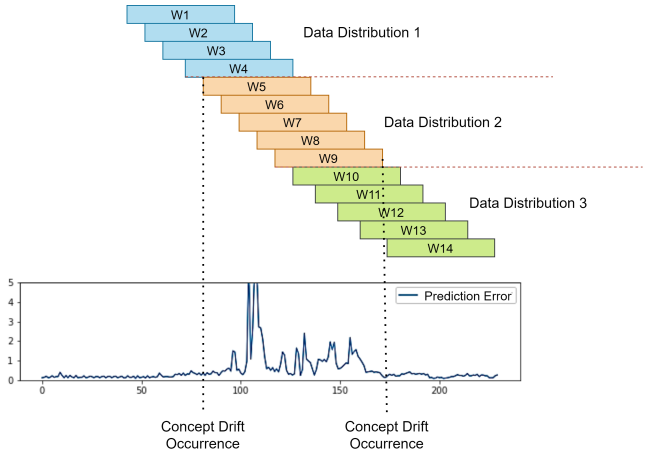

FIGURE 3. Sliding window technique

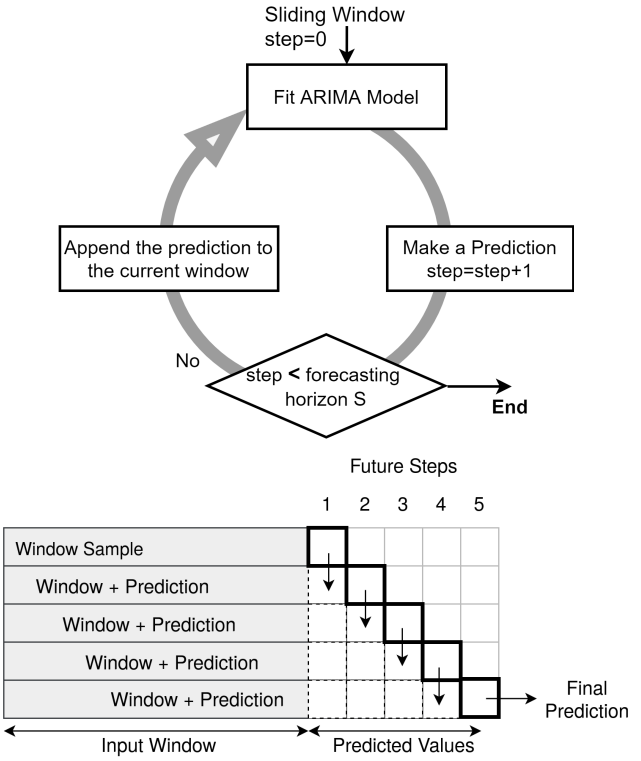

FIGURE 4. Rolling ARIMA example with the forecasting horizon of S steps

B. PREDICTION MODELS

This subsection describes the two base learners for the proposed ensemble: Rolling ARIMA and Online Adaptive RNN.

1) Rolling ARIMA

As described in Section IV-A, data from smart meters are transformed into overlapping windows and consecutive windows are placed together into batches which proceed to the prediction models. Therefore, ARIMA should be trained over a batch rather than a single window. Although the conventional ARIMA could be fitted over every window in a batch, this cannot take advantage of windows adjacency.

Consequently, the proposed ensemble uses a rolling ARIMA, an incremental ARIMA model trained in a rolling fashion such that to obtain the forecast S steps ahead, the model is fit on the window data and the prediction from $(S-1)$ steps. As seen in Fig. 4, the loop of fitting the model, predicting, and appending is repeated S times to obtain the prediction S steps ahead. As shown in Algorithm 1, the ARIMA model is fit to the window (line 4), and the model is used to predict one step ahead (line 5). Next, the obtained prediction is appended to the window (line 6) and ARIMA is fit on this expanded window (line 4). The process of fitting, predicting, and appending is repeated to obtain the prediction S time steps ahead. To acquire a stream of load predictions, the process is repeated for each window in the batch.

Algorithm 1: Rolling ARIMA

Input: Window: w, Forecast steps: S

Initialization: $model$=$ARIMA(p, d, q)$

$d \leftarrow w$

for *n=1,2,3,...,S* **do**

  $model \leftarrow fitARIMA(d)$

  $p \leftarrow model.predict(d)$

  $d \leftarrow d.append(p)$

2) Online Adaptive RNN

Online Adaptive RNN [9] is a deep learning model that can learn from data streams by updating the model as data become available. The overview of the model is shown in Fig. 5 [9]. The model consists of three modules *Normalizatio*, *BN-RNN*, and *Tuning*.

Normalization module: As this is a neural network-based technique, it requires normalization as an additional pre-possessing step to bring all features to a common scale, remove large feature dominance, and improve convergence. However, the traditional normalization is not possible because all data is not available at the start of the training and the normalization must be done as data arrive. This is addressed with Incremental Min-Max Normalization [9]. In this method, the largest and the smallest values of the features are tracked with $currentMax$ and $currentMin$, and updated as needed with the arrival of new data. The current batch is normalized using Min-Max normalization with the current values of $currentMax$ and $currentMin$ as follows:

$$\hat{x} = \frac{x - currentMin(x)}{currentMax(x) - currentMax(x)} \tag{10}$$

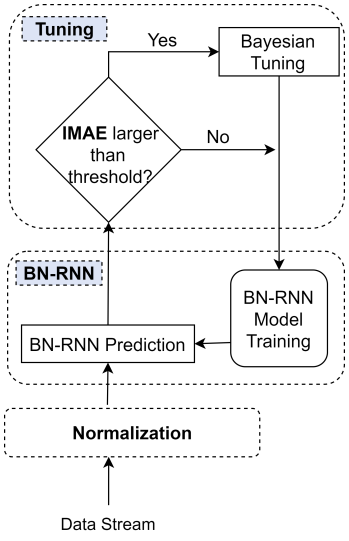

FIGURE 5. Online Adaptive RNN model

where x is the original feature value, $currentMin(x)$ and $currentMax(x)$ are the minimum and maximum of that feature from the beginning of the data stream, and $\hat{x}$ is the normalized value.

BN-RNN module: Online Adaptive RNN is empowered by Batch Normalized Recurrent Neural Networks (BN-RNN) [37] as the core learner. The batch normalization in BN-RNN indicates that the outputs of activation functions in the inner layers are normalized before passing them to the next layers. In RNNs, the batch normalization improves convergence and reduces training time. The BN-RNN is trained with the current batch and a limited number of epochs to prevent overfitting to the current batch [9].

Tuning module: This module is responsible for tracking the performance of the model and tuning it when needed. The predictions obtained by the BN-RNN module are passed to the tuning module which then uses the Incremental Mean Absolute Error (IMAE) to determine if tuning is needed. IMAE is calculated as follows:

$$IMAE_b = \frac{IMAE_{b-1} + MAE_b}{b} \tag{11}$$

where b is the current batch index, MAE_b is the MAE error for batch b, and $IMAE_{b-1}$ is the IMEA after batch $b-1$.

If IMAE starts to increase, a Bayesian tuning mechanism is activated to adjust the model hyperparameters before continuing to the next round of training. Note that this assessment accrues after the actual values for batch b become available [9]. If there is no significant change in IMEA, the training continues with the current hyperparameters.

C. ENSEMBLER

The ensembler module, Fig. 2, is responsible for combining the prediction from constituent base models, in this case, Rolling ARIMA and Online Adaptive RNN, to obtain the final prediction. In general, an ensemble model is expected to have higher accuracy in comparison to the single algorithm because of its generalization abilities [38]. In the ARIMA-RNN ensemble, the prediction models, Online Adaptive RNN and Rolling ARIMA are trained with each batch and then used to get the respective prediction values. While ARIMA hyperparameters (e.g., the degree of differencing, the order of the autoregressive and moving-average model) remain the same, RNN hyperparameters are optimized as new batches arrive.

As shown in Fig. 2, the final ensemble prediction is a combination of these individual predictions. The base models (RNN and ARIMA) prediction errors from the previous batch are indicators of how good each individual learner is at a specific time in the sequence, and therefore, these errors are considered as a criterion for generating the final prediction. Four ways of aggregating the final prediction are considered: average, weighted average, squared weighted average, and model switching. Fig. 6 illustrates the four aggregation techniques while the details are described in the following subsections.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

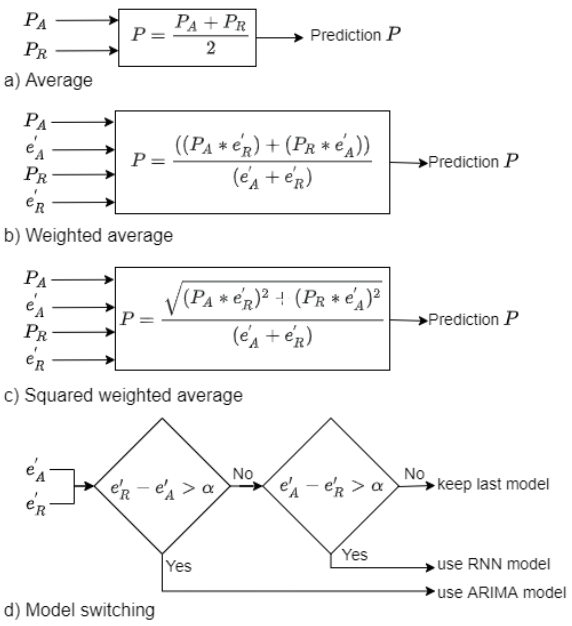

FIGURE 6. Aggregation techniques for the ARIM-RNN ensemble

1) Average

This is a simple average method belonging to the category of bagging ensemble approaches. The final prediction is the average of the base learner predictions, in our case Rolling ARIMA and Online Adaptive RNN. Equal weights of 0.5 are given to both models, and the final prediction P_{E1} is calculated as:

$$P_{E1} = \frac{P_A + P_R}{2} \tag{12}$$

where P_A and P_R are Rolling ARIMA and Online Adaptive RNN predictions.

2) Weighted Average

This is an adaptive boosting ensemble approach which uses Mean Absolute Error (MAE) of the base learners in the batch $n-1$ to determine the weights w_A and w_R given to the base learners predictions in the batch n. It is expected that the model with higher accuracy on the last batch should be given more weight on the current batch. The ARIMA weight w_A and Online Adaptive RNN weight w_R are calculated as follows:

$$w_A = 1 - \frac{e'_A}{e'_A + e'_R} \tag{13}$$

$$w_R = 1 - \frac{e'_R}{e'_A + e'_R} \rightarrow w_R = 1 - w_A \tag{14}$$

where e'_A and e'_R are the previous batch MAE errors for Rolling ARIMA and Online Adaptive RNN.

These weights ensure that the higher accuracy model from the last batch has more influence on the final prediction P_{E2}:

$$\begin{aligned} P_{E2} &= (P_A * w_A) + (P_R * w_R) \\ &= \frac{((P_A * e'_R) + (P_R * e'_A))}{(e'_A + e'_R)} \end{aligned} \tag{15}$$

3) Squared weighted average

Like the weighted average, the squared weighted average uses the error obtained by each base learner on the previous batch to determine the learners' impact on the current batch. In the weighted average, the learner impact is inversely proportional to the error while the squared weighted average approach increases the impact of the better model by squaring the weighted prediction. The final prediction is calculated as follows:

$$\begin{aligned} P_{E3} &= \sqrt{((P_A * w_A)^2 + (P_R * w_R)^2)} \\ &= \frac{\sqrt{(P_A * e'_R)^2 + (P_R * e'_A)^2}}{(e'_A + e'_R)} \end{aligned} \tag{16}$$

4) Model switching

The basic idea behind this approach is that the model that performed better on the batch $n-1$ has a high probability to perform better on the batch n. Therefore, this approach switches between the two models based on their performance on the last batch. This way, the weaker model is removed from the prediction. When there is no concept drift, it is expected that Online Adaptive RNN will perform better due to its ability to model complex patterns, and consequently, this model will be chosen for the following batch. In the presence of concept drift, Rolling ARIMA may perform better, and thus will be selected as the model for the final prediction.

The threshold α has been added to prevent switching between the models on a minimal difference in error. If on batch $n-1$, the accuracy of one model is higher than the accuracy of the other model by more than a factor of α, the better model is used for batch n. Otherwise, the same model is used on batch n as on $n-1$. The final prediction is as follows:

$$P_{E4} = \begin{cases} P_A, & \text{if } e'_R - e'_A > \alpha \\ P_R, & \text{if } e'_A - e'_R > \alpha \\ \text{keep last model}, & \text{otherwise} \end{cases} \tag{17}$$

where e'_A and e'_R are errors of ARIMA and RNN on the previous batch. The threshold α ensures that the model is not switched on minimal differences in error as that could indicate a false switch alarm and may not denote a concept drift. The value of α is determined through experiments as shown in Subsection V-B3.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

V. EVALUATION

London Hydro, a local electrical distribution utility involved with this project, developed the first Green Button Connect My Data (CMD) environment to provide secured sharing of energy data with the consumer's consent. Enhanced load forecasting will help London Hydro to increase return on investment from the smart meter infrastructure and will demonstrate the value of sharing smart meter data with 3rd parties through a secure Green Button platform. The proposed ARIMA-RNN ensemble was evaluated on proprietary real-world data from four residential consumers obtained through the CMD platform. Each consumer's dataset contained hourly energy consumption for three years, for a total of 25,560 readings. Additional features such as the day of the year, the hour of the day, and the day of the week, were devised from the load reading date/time to assist with modelling daily and weekly patterns. Weather-related features were also added including temperature, wind speed and direction, pressure, and humidity. The complete list of features, together with an example of their value, is shown in Table 1. All non-numeric features are converted to numbers with one-hot encoding. Energy consumption is the target variable. Note that we used many features and relied on the ability of the deep learning model to extract relevant features. To examine the differences between the four houses in respect to the temporal patterns present in data, concept drift analysis is conducted first. Next, the accuracy of Rolling ARIMA, Online Adaptive RNN, and the ensemble is examined. Finally, different load forecasting models are compared and statistical significance is examined.

A. CONCEPT DRIFT ANALYSIS

The presence of the concept drift degrades the performance of the offline ML algorithms and may even make them unusable. Although online ML techniques for load forecasting have an advantage as they can adapt to changes in data patterns, the examination of the impact of the concept drift on the load forecasting accuracy has been limited [9]. Consequently, here we first investigate the four houses with respect to the presence of the concept drift.

Concept drift detection techniques work on different principles and detect different types of drifts. Therefore, to examine different drifts, three techniques have been used: Drift-Detection Method (DDM) [39], ADaptive-WINdowing (ADWIN) [40] and Page-Hinkley [41].

DDM, one of the commonly referenced methods, is based on the Binomial distribution giving the probability for the random variable representing the error. A significant increase in the online error indicates drift occurrence. In ADWIN, the model has an adaptive window w of a variable size: the window grows when there is no change in the statistical properties of data and shrinks otherwise. If there are sub-windows of w with distinct properties, the drift is detected. Finally, Page-Hinkley keeps track of the cumulative difference between the time series values and their current mean. This cumulative sum is compared against its minimum to detect the concept drift.

Fig. 7 shows the number of concept drifts detected by each of the four algorithms for the four houses. It can be observed that ADWIN detected a higher number of drifts for house one than for house two, while DDM did not detect any drifts for house one and detected five for house two. Nevertheless, with most algorithms, houses one and two exhibit a fewer number of drifts than houses three and four; thus, we will refer to houses one and two as Low 1 and Low 2 and to houses three and four as High 1 and High 2.

Figures 8 and 9 show detected concept drifts for homes Low 2 and High 2. Drift information is overlapped with actual load data for those houses. DDM detects concept drift occurrence along with the duration of the drift: sections with the same colour indicate the data with similar distributions. In contrast, ADWIN and Page-Hinkley only indicate the time at which the concept drift starts: the peak location indicates drift occurrence while the height of the peak depicts the actual load value at that point. It can be observed that each algorithm detected a higher number of drifts for house High 2 than for Low 2.

The DDM algorithm especially highlights this difference by many differently coloured areas for house High 2 indi-

TABLE 1. Features with data examples

Feature	Example
Year	2016
Month	11
The Day of the Month	24
The Hour of the Day	16
The Week Number	23
The Day of the Week	4
Season	Spring
Holiday	True or False
Weather Condition	Rainy
Temperature	$-14.2C°$
Dew Point Temperature	$-16.4C°$
Relative Humidity	71%
Wind Direction	29°
Wind Wind Speed	21 mph
Visibility	24.1 meter
Energy Consumption	0.38 KW

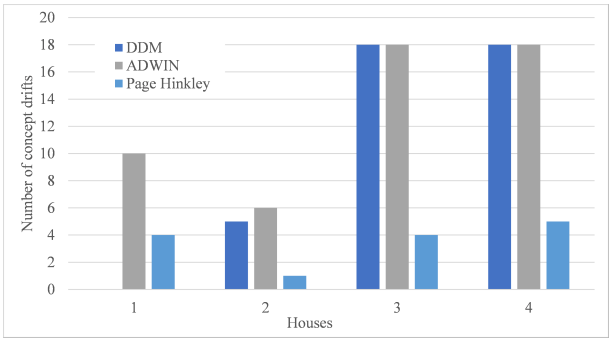

FIGURE 7. The number of concept drifts detected by DDM, ADWIN and Page-Hinkley for the four houses

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

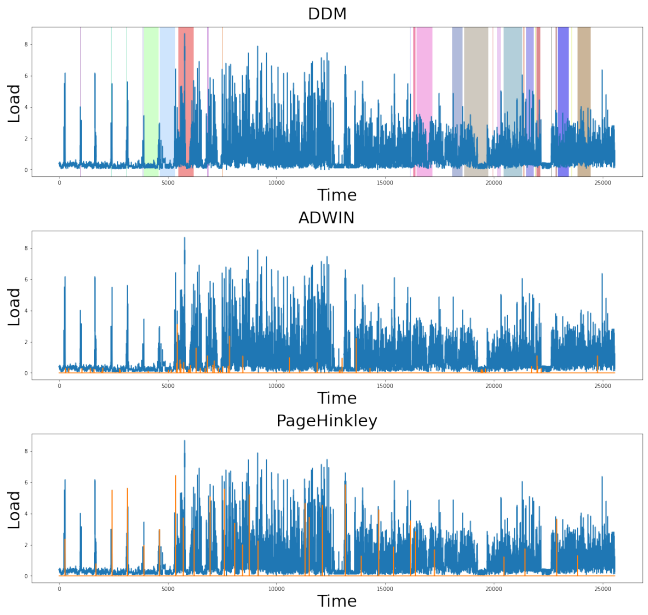

FIGURE 8. Drifts detected by DDM, ADWIN and PageHinkley - house High 2

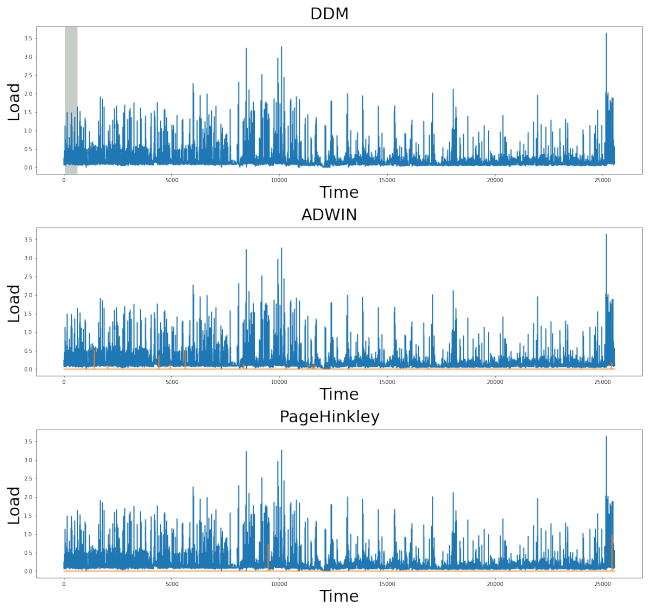

FIGURE 9. Drifts detected by DDM, ADWIN and PageHinkley - house Low 2

cating diverse distributions and concept drifts throughout the time series. These concept drifts create challenges for online learning as the model must perpetually change to capture new data patterns.

B. ACCURACY ANALYSIS FOR ROLLING ARIMA, ONLINE ADAPTIVE RNN, AND ARIMA-RNN ENSEMBLE

Offline learning models are static and thus can be evaluated on a static holdout set containing samples that were not used for model training. In contrast, the assessment of the online models is more complex as the models change over time by learning from continuously arriving data. Still, the out-of-sample evaluation approach must be used to achieve realistic estimates. We are using the holdout-like technique on every window: W consecutive time steps with the corresponding load, weather, and date/time features constitute the independent variables and the load to predict is the dependent variable. The sample is first used as a test sample by predicting the load value and comparing it to the actual consumption. Next, the same sample is used for the training. The process is repeated for the next window and so on. This is similar to prequential evaluation [9], but in our study, it is used with the windowing approach.

With traditional machine learning, the larger the training data set, the higher the model accuracy. For load forecasting with window sliding techniques, a large window size typically leads to higher accuracy [13]. However, the model performance under the concept drift depends on discounting or even removing the insights obtained from the data collected before the concept drift [42]. Thus, there are opposing needs to increase window length for accuracy increase and to reduce the window length for better concept drift adaptation. The load prediction under the concept drift requires a trade-off between the prediction accuracy on the non-drifting signal and the accuracy while adapting to the new concepts. Consequently, this subsection examines the impact of windows size on the accuracy of Rolling ARIMA, Online Adaptive RNN, and the proposed ARIMA-RNN Ensemble.

1) Rolling ARIMA

As the window size increases, ARIMA has more data points to fit the model, and therefore, it is expected that the prediction accuracy will increase. However, as the window size increases further, the ability of the model to adapt to the new concepts may decrease as the adaptation may take longer because of the presence of older samples in the window.

Experiments were conducted with window sizes from 100 to 600 time steps, with increments of 100. The Mean Squared Error (MSE) for the four houses as well as the average MSE are shown in Fig. 10. As expected, with the increase of window size from 100 to 200, the error decreases for most houses. With the further increase of the window size, the error remains the same or marginally increases. Consequently, window size 200 was selected for use in the ensemble.

House Low 1 achieved lower errors than the other houses for all window sizes. This could be explained by the fact that this home has fewer concept drifts than other homes. However, house Low 2 had the highest errors of the four homes despite the low concept drift presence. A possible reason is the presence of complex dependencies in data not captured by the ARIMA model.

Fig. 11 demonstrates the ARIMA performance for an example concept drift: the two graphs show MSE for ARIMA with window sizes 200 and 600 overlapped with the coloured region indicating the concept drift according to DDM. For both window sizes, the errors are low before and after the concept drift, and the highest spikes are observed during the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

concept drift. The height of the spikes during the drift is slightly lower for window size 200 than for 600.

2) Online Adaptive RNN

Similar to Rolling ARIMA, Online Adaptive RNN accuracy is affected by the window size, and therefore, the experiments for Online Adaptive RNN were also conducted with window sizes from 100 to 600 time steps, with increments of 100. The architecture of the RNN remains the same through all experiments: one LSTM layer with 64 hidden units. The learning rate is tuned online, whenever IMAE drops, as described in Subsection IV-B2.

The results are shown on Fig. 12. Whereas with ARIMA, the error for all houses decreased with increasing the window size from 100 to 200 (Fig. 10), for Online Adaptive RNN, the error remained almost the same (Fig. 12). The average error only exhibited small oscillations, with the lowest value achieved for window size 100. Thus, the window size 100 was used for the Online Adaptive RNN in the ensemble experiments.

As in ARIMA experiments, house Low 2 achieved lower error irrelevant of the window size. Whereas with ARIMA, house Low 1 had the highest error rates (Fig. 10), with Online Adaptive RNN, house Low 1 achieved lower errors than house High 2 (Fig. 12). This can be explained by the fact that Online Adaptive RNN is capable of capturing more complex patterns than Rolling ARIMA. Overall, errors are much lower for Online Adaptive RNN than for Rolling ARIMA.

An example of Online Adaptive RNN performance in presence of the concept drift is shown in Fig. 13: graphs show MSE values for house High 2, window sizes 200 and 300, with the concept drift indicated with the coloured region. This house and window sizes were chosen for illustration as its forecasting error increased significantly when window size was increased from 200 to 300 as observed from Fig. 12. At the start of the concept drift, the spikes are higher for the window size of 300 than for 200. With a large window size, the model needs more time to adjust to the concept drift. In the later part of the concept drift, errors are lower for the window size of 200.

Online Adaptive RNN hyperparameters were determined by trial and error starting from the tuned model presented by Fekri *et al.* [9]. Note that the learning rate, the most important RNN hyperparameter [9], is tuned online by Bayesian optimization.

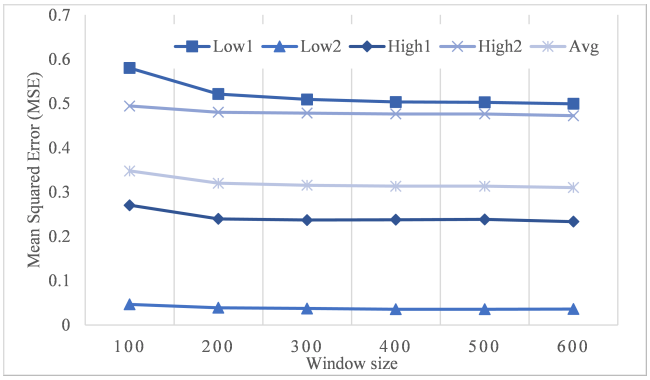

FIGURE 10. Impact of the window size on MSE for ARIMA

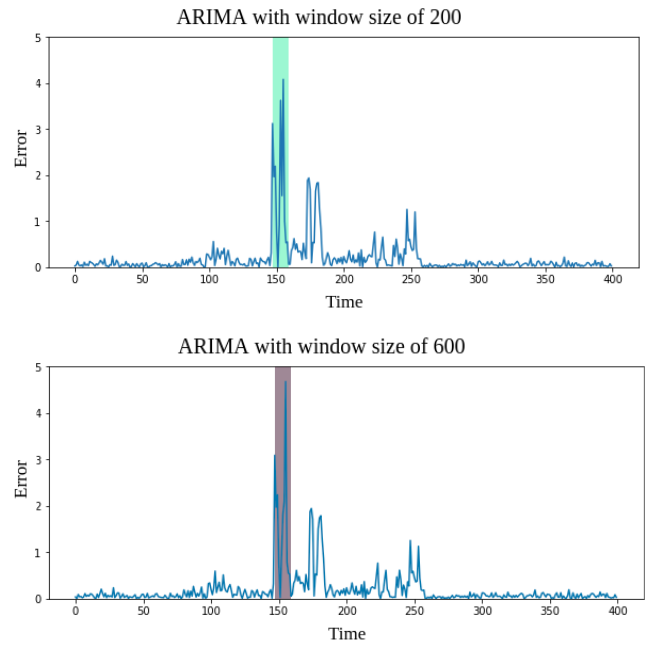

FIGURE 11. Examples of Rolling ARIMA errors for window sizes 200 and 600

3) Ensemble ARIMA-RNN

Ensemble ARIMA-RNN used ARIMA and Online Adaptive RNN with parameters determined individually as described in the previous two subsections. The only ensemble variant with additional parameters is the model switching aggregation which uses threshold α for switching between models. As switching the models too frequently may result in the degradation of the overall ensemble performance; this subsection examines the impact of the α threshold.

The experiments were conducted with α values from 1 to

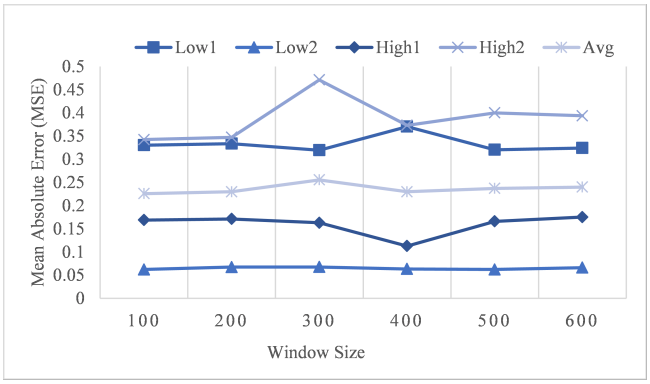

FIGURE 12. Impact of the window size on MSE for Online Adaptive RNN

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

3, with increments of 0.25: the results are shown in Fig. 14. As with Rolling ARIMA (Fig. 10) and Online Adaptive RNN (Fig. 12), for the ensemble, house Low 2 showed the lowest errors, and houses Low 1 and High 2 exhibited the highest errors. The average error decreased as α increased: the homes with higher errors exhibiting a sharper decrease in errors than the homes with lower errors. This confirms that between the participating models, Rolling ARIMA and Online Adaptive RNN, on small differences in the prediction accuracy leads to performance degradation. The α value of 3 was used in the remaining experiments.

C. MODEL COMPARISON

This subsection compares the proposed ARIMA-RNN ensemble with each of its constituent learning algorithms: Rolling ARIMA and Online Adaptive RNN. Ferki *et al.* showed that Online Adaptive RNN achieves better results than state of the art online and offline models [9] including deep models such as LSTM. Here we show that, in presence of the concept drift, the Online Adaptive RNN results can be further improved by combining Online Adaptive RNN with Rolling ARIMA. Moreover, we compare ensemble models with Online Linear Regression [43] and Online Bagging Regression [44], investigate different aggregation techniques, and examine the behaviour of the models under different levels of concept drift.

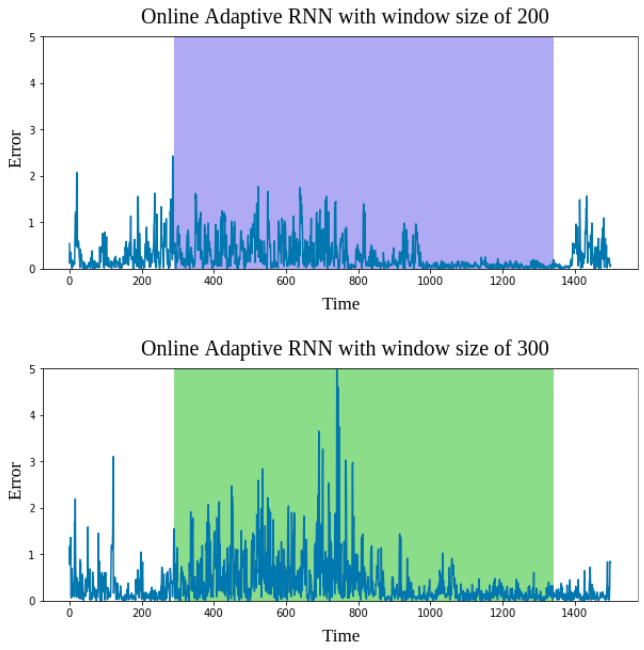

FIGURE 13. Examples of Online Adaptive RNN errors for window sizes 200 and 600

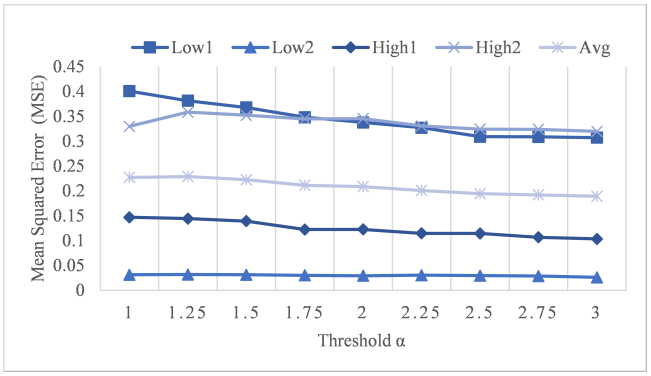

FIGURE 14. Impact of α value on ensemble MSE

Figures 15 and 16 compare Rolling ARIMA, Online Adaptive RNN, online linear regression, and online bagging regression with the four variants of the proposed ARIMA-RNN Ensemble based on the type of aggregation: average, weighted average, squared weighted average, and model switching. Fig. 15 shows MSE while Fig. 16 shows MAE for the four houses and the average among the four houses.

As seen from Fig. 15, the lowest average error is achieved by the ensemble method with simple averaging. This method also achieved the lowest error for houses Low 1, Low 2, and High 2, while for house High 1, simple and weighted average achieved almost the same accuracy. The lowest errors, irrelevant of the model, were achieved for house Low 2, and for this house, there is very little difference between Online Adaptive RNN and any ensemble model. However, for house Low 1, three ensemble variants achieve better accuracy than Online Adaptive RNN, with the best algorithm being the ensemble with simple averaging.

All approaches achieved the lowest errors for house Low 2, followed by High 1, High 2, and Low 1. When houses exhibit a higher presence of concept drift such as High 2, the performance of all algorithms degrades. Also, errors for house Low 1 were high for all algorithms what may be explained with high variability of data. The benefit of the proposed ensemble is the most evident for houses with higher errors, Low 1 and High 2, when the difference between the ensemble with simple averaging and Online Adaptive RNN increases.

Comparing MAEs among houses, Fig. 16, the ensemble with weighted average achieved overall the best results, but the difference from the Online Adaptive RNN is smaller than in the case of MSE. This is caused by the difference between the algorithms being highlighted with the squaring operation in MSE.

Fig. 17 shows an example of how each of the considered algorithms handles the concept drift. During the concept drift indicated by the coloured region, each of the algorithms exhibits spikes in the errors. The ensemble approaches show somewhat lower spikes during the concept drift, which corresponds to their better accuracy as observed from Fig. 15. Also, all algorithms experience some challenges between time steps 250 to 300 when errors increase without the presence of concept drift. Although, the DDM did not detect the presence of the concept drift during that segment, the increase of errors for all six algorithms indicates that there is a difference in data between the time steps 250 and 300.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

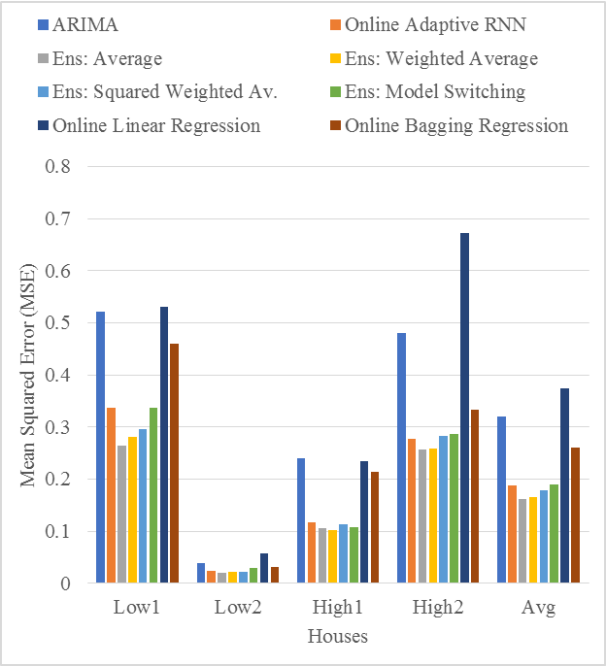

FIGURE 15. Comparison of different approaches in terms of MSE

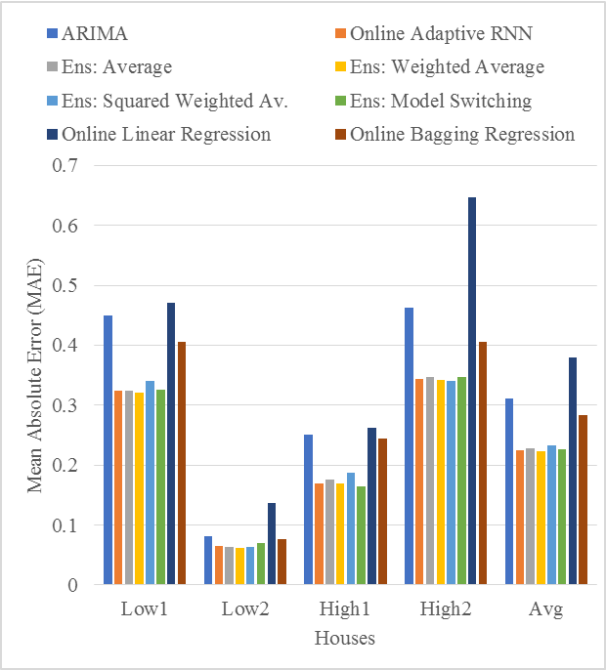

FIGURE 16. Comparison of different approaches in terms of MAE

D. STATISTICAL SIGNIFICANCE

The Diebold-Mariano test introduced in Section III-D is used to determine if the difference in forecasting performance between algorithms is significant. The null hypothesis is: the two algorithms are not significantly different. Figures 18-21 show DM values for each pair of algorithms, for houses Low1, Low2, High1, and High2.

As seen from Fig. 18 for house Low1, the DM values for

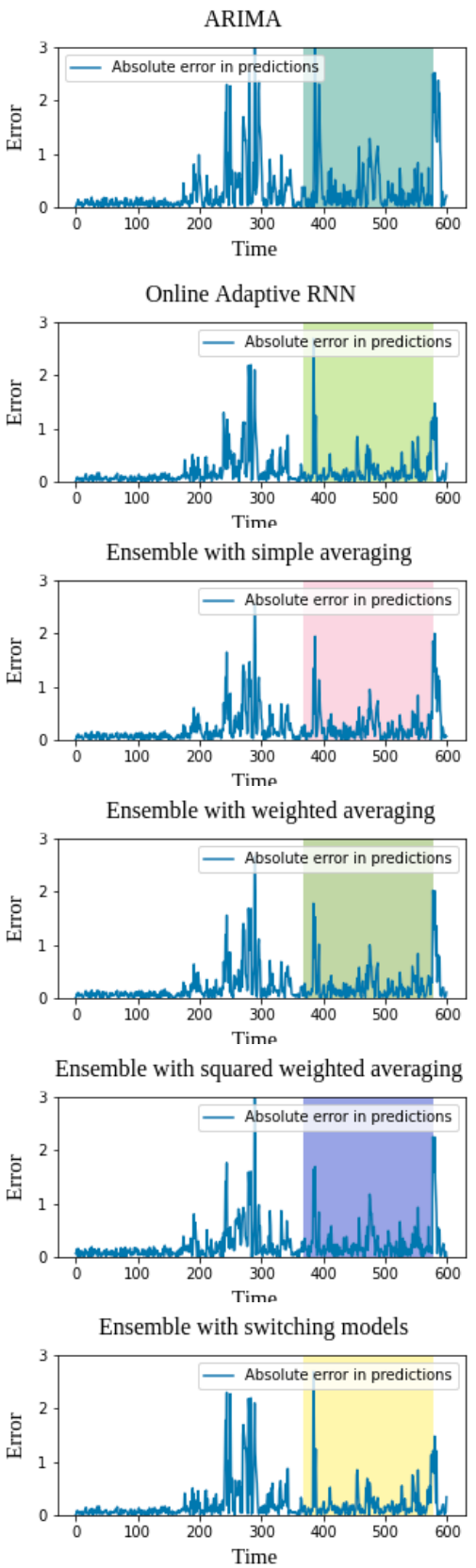

FIGURE 17. An example of errors observed during the concept drift

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3095420, IEEE Access

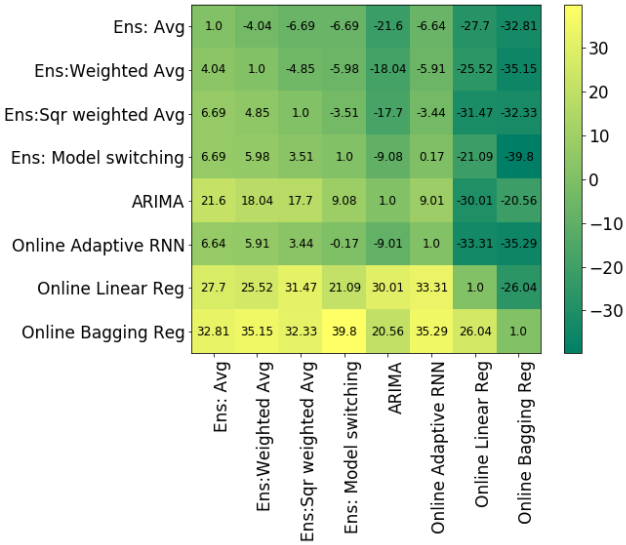

FIGURE 18. Diebold-Mariano test: DM values for house Low1

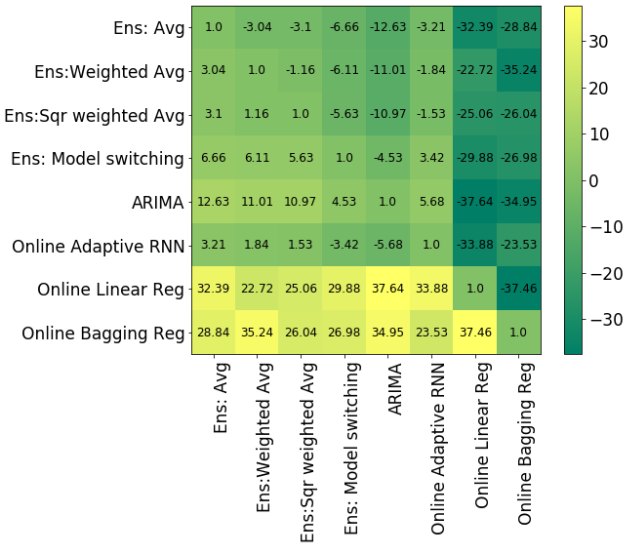

FIGURE 19. Diebold-Mariano test: DM values for house Low2

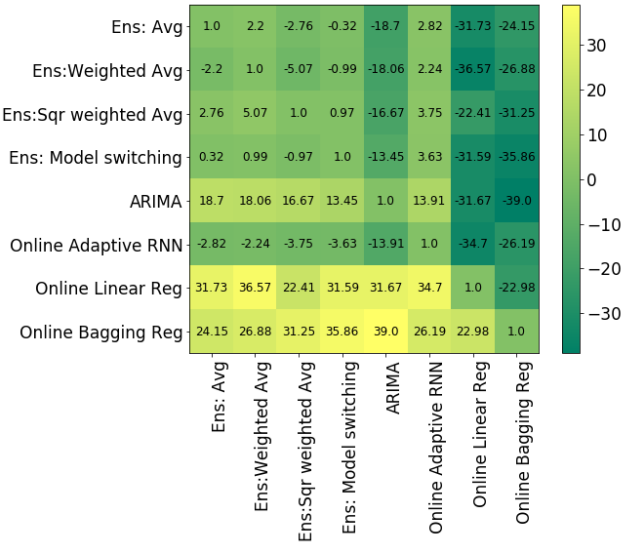

FIGURE 20. Diebold-Mariano test: DM values for house High1

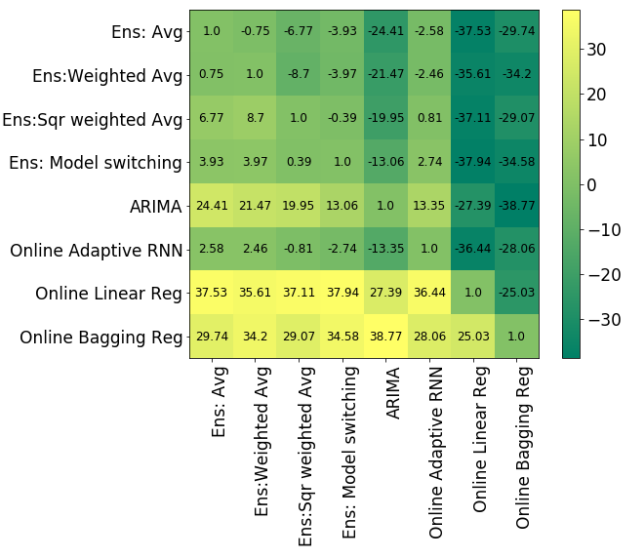

FIGURE 21. Diebold-Mariano test: DM values for house High2

all pairs, except for the pair {Ensemble Model Switching, Online Adaptive RNN}, are beyond the range [-1.96 1.96] indicating that those pairs of algorithms achieve significantly different forecasts.

Investigating DM values for house Low2 shown in Fig. 19, the pairs {Ensemble Weighted Average, Online Adaptive RNN}, {Ensemble Squared Weighted Average, Online Adaptive RNN}, and {Ensemble Squared Weighted Average, Ensemble Weighted Average} have the DM values in the range of [-1.96, 1.96]; thus, for those pairs, the null hypothesis is not rejected and the superiority of one algorithm over the other is not significant. However, for the remaining pairs of algorithms, the DV values are significant.

Fig. 20 shows the DM values for house High1. It can be observed that the DM values for pairs {Ensemble Model Switching, Ensemble Squared Weighted}, {Ensemble Model Switching, Ensemble Weighted Average}, and {Ensemble Model Switching, Ensemble Average} are in the range that cannot reject the null hypothesis. In other words, one algorithm is not statistically better than the other. However, all three algorithms involved in those pairs are the variants of the proposed ensemble and the other pairs have shown significant differences in their performance.

Finally, for house High2, as seen from Fig. 21, all approaches have shown significant differences in their performance except for pairs {Ensemble Squared Weighted Average, Online Adaptive RNN}, {Ensemble Squared Weighted Average, Ensemble Model Switching}, and {Ensemble

Weighted Average, Ensemble Average}.

The overall outcome of the Diebold-Mariano test indicates that the forecasts obtained by the proposed ensembles are significantly different from other approaches. The only algorithm that comes close to the proposed ensemble is Online Adaptive RNN: for house Low2, Online Adaptive RNN obtains similar forecasts to Ensemble Weighted Average and for house High2, results of Online Adaptive RNN are similar to Ensemble Squared Weighted Average. Nevertheless, for the remaining two houses, the results of the ensemble models are significantly different from Online Adaptive RNN making the ensemble overall better solution.

E. DISCUSSION

Load forecasting plays an essential role in smart grids because of its importance in estimating future demand, balancing production with consumption, designing demand-response initiatives, energy budgeting, and similar. This importance is reflected in a plethora of techniques proposed for load forecasting. Although these techniques have been achieving great accuracy, they are mostly static and cannot accommodate changing patterns. Whereas this may be acceptable when dealing with one or a few models and retraining them as needed, it becomes unfeasible when dealing with a large number of models (possibly corresponding to individual smart meters) or fast-changing data. It is imperative that we transition toward more dynamic, online models.

Even these online models can exhibit very different accuracy for different entities, such as the four houses considered in this study. As shown in figures 15 and 16, the accuracy can vary greatly even among houses, and one technique may not be achieving the very best results for each of the houses. This highlights the need to evaluate the proposed algorithms on different data streams.

Typically, load forecasting approaches are compared in terms of the average error such as MSE or MAE. As shown in Fig. 15, the proposed ensemble achieved the best overall average accuracy in terms of MSE. However, this average accuracy does not provide a complete picture of the algorithm's behaviour as the error may vary greatly for different time periods. This can be observed in figures 11, 13, and 17 for Rolling ARIMA, Online Adaptive RNN, and the proposed ARIMA-RNN ensemble when the error values spike during the concept drift and occasionally even outside the drift.

While many machine learning-based load forecasting studies have been published in recent years, almost all of them employ offline learning and, therefore, produce static models. In contrast, our Ensemble ARIMA-RNN is an online technique capable of adapting to new data as they arrive. Like our study, the work of Fekri *et al.* [9] demonstrated the importance of moving from offline to online learning in load forecasting. While Fekri *et al.* also consider concept drift, we improve the accuracy in presence of concept drift by employing an ensemble model.

To enable the use of load forecasting techniques on a large number of residential consumers (or smart meters) and for longer periods of time, the approaches must be able to adapt fast to changes in data. This study examines the proposed ensemble in respect to how it adapts to concept drift and illustrates the need to evaluate the load forecasting algorithms with respect to concept drift.

VI. CONCLUSION

Forecasting on a regional level as well as forecasting for schools and offices have been achieving high accuracy because data patterns are well defined and consistent. In contrast, residential electricity consumption data exhibits high variability and the presence of concept drift. Consequently, for such consumers, online learning is becoming paramount as the models must adapt to newly arriving data.

This paper proposes combing ARIMA and RNN for load forecasting under the concept drift. The RNN part is using Online Adaptive RNN to capture time dependencies and achieve online learning. The ARIMA component makes use of the rolling technique to improve the ensemble's adaptation to the concept drift. The experiments on four homes with different degrees of concept drift show that the proposed ensemble approach outperforms its underlying algorithms, Rolling ARIMA and Online Adaptive RNN. Moreover, the evaluation demonstrates the importance of examining the performance of the load forecasting techniques on different data sets, with different degrees of concept drift and the need to examine, not only the average error but also the errors occurring during data drifts.

As currently there are very few studies employing online learning techniques for load forecasting, these techniques need to be further examined in diverse scenarios such as long-term forecasting and aggregated load forecasting. Our study considered concept drift, and even investigated different degrees of concept drift; however, we did not quantify concept drift or the performance during the drifting period. Future work will investigate establishing techniques and metrics for evaluating the performance of algorithms under the presence of concept drift. Moreover, the neural network will be examined as a way of merging individual algorithms.

REFERENCES

[1] U.S. Energy Information Administration. International energy outlook 2019. https://www.eia.gov/todayinenergy/detail.php?id=41433, 2019.

[2] Ya Cheng, Usama Awan, Shabbir Ahmad, and Zhixiong Tan. How do technological innovation and fiscal decentralization affect the environment? A story of the fourth industrial revolution and sustainable growth. Technological Forecasting and Social Change, 162:120398, 2021.

[3] Ye Hong, Yingjie Zhou, Qibin Li, Wenzheng Xu, and Xiujuan Zheng. A deep learning method for short-term residential load forecasting in smart grid. IEEE Access, 8:55785–55797, 2020.

[4] Yi Wang, Qixin Chen, Tao Hong, and Chongqing Kang. Review of smart meter data analytics: Applications, methodologies, and challenges. IEEE Transactions on Smart Grid, 10(3):3125–148, 2018.

[5] Davoud Gholamiangonabadi, Nikita Kiselov, and Katarina Grolinger. Deep neural networks for human activity recognition with wearable sensors: Leave-one-subject-out cross-validation for model selection. IEEE Access, 8:133982–133994, 2020.

[6] Alexandra L'heureux, Katarina Grolinger, Hany F Elyamany, and Miriam AM Capretz. Machine learning with big data: Challenges and approaches. IEEE Access, 5:7776–7797, 2017.

[7] Indrė Žliobaitė, Mykola Pechenizkiy, and Joao Gama. An overview of concept drift applications. In Big data analysis: new algorithms for a new society, pages 91–114. 2016.

[8] Heitor Murilo Gomes, Jesse Read, Albert Bifet, Jean Paul Barddal, and João Gama. Machine learning for streaming data: state of the art, challenges, and opportunities. ACM SIGKDD Explorations Newsletter, 21(2):6–22, 2019.

[9] Mohammad Navid Fekri, Harsh Patel, Katarina Grolinger, and Vinay Sharma. Deep learning for load forecasting with smart meter data: Online adaptive recurrent neural network. Applied Energy, 282:116177, 2020.

[10] Tanveer Ahmad, Hongcai Zhang, and Biao Yan. A review on renewable energy and electricity requirement forecasting models for smart grid and buildings. Sustainable Cities and Society, 55(102052), 2020.

[11] Abdullah Al Mamun, Md Sohel, Naeem Mohammad, Md Samiul Haque Sunny, Debopriya Roy Dipta, and Eklas Hossain. A comprehensive review of the load forecasting techniques using single and hybrid predictive models. IEEE Access, 8:134911–134939, 2020.

[12] Gholamreza Memarzadeh and Farshid Keynia. Short-term electricity load and price forecasting by a new optimal LSTM-NN based prediction algorithm. Electric Power Systems Research, 192:106995, 2021.

[13] Ljubisa Sehovac and Katarina Grolinger. Deep learning for load forecasting: Sequence to sequence recurrent neural networks with attention. IEEE Access, 8:36411–36426, 2020.

[14] Hosein Eskandari, Maryam Imani, and Mohsen Parsa Moghaddam. Convolutional and recurrent neural network based model for short-term load forecasting. Electric Power Systems Research, 195:107173, 2021.

[15] Yifang Tian, Ljubisa Sehovac, and Katarina Grolinger. Similarity-based chained transfer learning for energy forecasting with big data. IEEE Access, 7:139895–139908, 2019.

[16] Xianlun Tang, Yuyan Dai, Qing Liu, Xiaoyuan Dang, and Jin Xu. Application of bidirectional recurrent neural network combined with deep belief network in short-term load forecasting. IEEE Access, 7:160660–160670, 2019.

[17] Xianlun Tang, Yuyan Dai, Ting Wang, and Yingjie Chen. Short-term power load forecasting based on multi-layer bidirectional recurrent neural network. IET Generation, Transmission & Distribution, 13(17):3847–3854, 2019.

[18] Pin Lv, Song Liu, Wenbing Yu, Shuquan Zheng, and Jing Lv. EGA-STLF: A hybrid short-term load forecasting model. IEEE Access, 8:31742–31752, 2020.

[19] Heng Shi, Minghao Xu, and Ran Li. Deep learning for household load forecasting—a novel pooling deep RNN. IEEE Transactions on Smart Grid, 9(5):5271–5280, 2017.

[20] Ming Dong and Lukas Grumbach. A hybrid distribution feeder long-term load forecasting method based on sequence prediction. IEEE Transactions on Smart Grid, 11(1):470–482, 2019.

[21] Lingxiao Wang, Shiwen Mao, Bogdan M Wilamowski, and RM Nelms. Ensemble learning for load forecasting. IEEE Transactions on Green Communications and Networking, 4(2):616–628, 2020.

[22] Onat Gungor, Jake Garnier, Tajana S Rosing, and Baris Aksanli. Lenard: Lightweight ensemble learner for medium-term electricity consumption prediction. In 2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, pages 1–6, 2020.

[23] Mohamed Massaoudi, Shady S Refaat, Ines Chihi, Mohamed Trabelsi, Fakhreddine S Oueslati, and Haitham Abu-Rub. A novel stacked generalization ensemble-based hybrid lgbm-xgb-mlp model for short-term load forecasting. Energy, 214:118874, 2021.

[24] Javier J Sánchez-Medina, Juan Antonio Guerra-Montenegro, David Sánchez-Rodríguez, Itziar G Alonso-González, and Juan L Navarro-Mesa. Data stream mining applied to maximum wind forecasting in the Canary Islands. Sensors, 19(10), 2019.

[25] Tian Guo, Zhao Xu, Xin Yao, Haifeng Chen, Karl Aberer, and Koichi Funaya. Robust online time series prediction with recurrent neural networks. In IEEE International Conference on Data Science and Advanced Analytics, pages 816–825, 2016.

[26] Sandeep Madireddy, Prasanna Balaprakash, Philip Carns, Robert Latham, Glenn K Lockwood, Robert Ross, Shane Snyder, and Stefan M Wild. Adaptive learning for concept drift in application performance modeling. In International Conference on Parallel Processing, pages 1–11, 2019.

[27] Julian Vexler and Stefan Kramer. Integrating LSTMs with online density estimation for the probabilistic forecast of energy consumption. In International Conference on Discovery Science, pages 533–543, 2019.

[28] Fan Liang, William Grant Hatcher, Guobin Xu, James Nguyen, Weixian Liao, and Wei Yu. Towards online deep learning-based energy forecasting. In International Conference on Computer Communication and Networks, pages 1–9, 2019.

[29] Leandro Von Krannichfeldt, Yi Wang, and Gabriela Hug. Online ensemble learning for load forecasting. IEEE Transactions on Power Systems, 36(1), 2020.

[30] B Shiva Prakash, KV Sanjeev, Ramesh Prakash, and K Chandrasekaran. A survey on recurrent neural network architectures for sequential learning. In Soft Computing for Problem Solving, pages 57–66. 2019.

[31] Javier Contreras, Rosario Espinola, Francisco J Nogales, and Antonio J Conejo. Arima models to predict next-day electricity prices. IEEE transactions on power systems, 18(3):1014–1020, 2003.

[32] Ashfaqur Rahman and Brijesh Verma. A novel ensemble classifier approach using weak classifier learning on overlapping clusters. In The 2010 international joint conference on neural networks, pages 1–7, 2010.

[33] Harris Drucker, Corinna Cortes, Lawrence D Jackel, Yann LeCun, and Vladimir Vapnik. Boosting and other ensemble methods. Neural Computation, 6(6):1289–1301, 1994.

[34] Magdalena Graczyk, Tadeusz Lasota, Bogdan Trawiński, and Krzysztof Trawiński. Comparison of bagging, boosting and stacking ensembles applied to real estate appraisal. In Asian conference on intelligent information and database systems, pages 340–350, 2010.

[35] Hao Chen, Qiulan Wan, and Yurong Wang. Refined diebold-mariano test methods for the evaluation of wind power forecasting models. Energies, 7(7):4185–4198, 2014.

[36] Peng Xu, Muhammad Aamir, Ani Shabri, Muhammad Ishaq, Adnan Aslam, and Li Li. A new approach for reconstruction of imfs of decomposition and ensemble model for forecasting crude oil prices. Mathematical Problems in Engineering, 2020, 2020.

[37] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167, 2015.

[38] Lei Lu, Xiaoqin Zeng, Shengli Wu, and Shuiming Zhong. A novel ensemble approach for improving generalization ability of neural networks. In International Conference on Intelligent Data Engineering and Automated Learning, pages 164–171, 2008.

[39] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In Brazilian symposium on artificial intelligence, pages 286–295, 2004.

[40] Isvani Frías-Blanco, José del Campo-Ávila, Gonzalo Ramos-Jimenez, Rafael Morales-Bueno, Agustín Ortiz-Díaz, and Yailé Caballero-Mota. Online and non-parametric drift detection methods based on Hoeffding's bounds. IEEE Transactions on Knowledge and Data Engineering, 27(3):810–823, 2014.

[41] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. ACM computing surveys, 46(4):1–37, 2014.

[42] Anjin Liu, Guangquan Zhang, and Jie Lu. Fuzzy time windowing for gradual concept drift adaptation. In 2017 IEEE International Conference on Fuzzy Systems, pages 1–6, 2017.

[43] Max Halford, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, and Adil Zouitine. creme, a Python library for online machine learning, 2019.

[44] Nikunj C Oza. Online bagging and boosting. In IEEE international conference on systems, man and cybernetics, volume 3, pages 2340–2345, 2005.

RASHPINDER KAUR JAGAIT received her B.Eng. from PEC University of Technology, India and M.Eng. in Software engineering, collaborative specialization in artificial intelligence (CSAI), from Western University, London, ON, Canada. She has been a software developer in Goldman Sachs, India and is currently working as a Developer at GroupBy Inc., Canada. Her current research interests are machine learning, deep learning, data analytics, and Big Data.

MOHAMMAD NAVID FEKRI is currently pursuing a Ph.D. degree in software engineering at Western University, Canada. He received the B.Sc. degree in software engineering from Isfahan University, Iran, and the M.Sc. degree in Artificial Intelligence from Iran University of Science and Technology, Iran. His current research focuses on online machine learning, federated learning, and IoT.

KATARINA GROLINGER received the B.Sc. and M.Sc. degrees in mechanical engineering from the University of Zagreb, Croatia, and the M.Eng. and Ph.D. degrees in software engineering from Western University, Canada. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Western University. She has been involved in the software engineering area in academia and industry, for over 20 years. Her current research interests include machine learning, sensor data analytics, data management, and the IoT.

SYED MIR is the Vice President of Corporate Services and CIO at London Hydro, where he is responsible for Customer Services, Meter Services and Information Technology. He is also Chair of Green Button Alliance. He has vast experience and has served in various roles in several companies in the energy sector. Mr. Mir holds a BSc in Computer Science from Western University. His research interests are cloud computing, mobility, and smart grids.