Adaptive load forecasting using reinforcement learning with database technology

Chee Keong Wee & Richi Nayak

To cite this article: Chee Keong Wee & Richi Nayak (2019) Adaptive load forecasting using reinforcement learning with database technology, Journal of Information and Telecommunication, 3:3, 381-399, DOI: 10.1080/24751839.2019.1596470

To link to this article: https://doi.org/10.1080/24751839.2019.1596470
Adaptive load forecasting using reinforcement learning with database technology

Chee Keong Wee and Richi Nayak

Science and Engineering Faculty, Queensland University of Technology, Brisbane, Australia

ABSTRACT
Load forecasting is an essential operation in the power utility industry. However, a common challenge is faced for adjusting forecasting models to fit the need for substations’ load prediction as well as minimizing expenditure in IT resources for repurposing these forecasting models to bigger datasets. The goal of this paper is to propose a novel solution that is responsive to these demands through the integration of reinforcement learning with load forecasting on existing database technology. To deal with the varying accuracy of the forecasting models on different substations’ loads, the proposed solution compares and uses the best models and recalibrate them iteratively by comparing the model’s prediction against the actual load data. As shown in empirical analysis, the solution interacts with the environment and performs the optimum forecasting routine intuitively.

ARTICLE HISTORY
Received 12 June 2018
Accepted 14 March 2019

KEYWORDS
Forecasting; reinforcement learning; electricity utility; database technology

1. Introduction
This paper is an extended version of the paper that was published in ACIIDS2018 conference (https://doi.org/10.1007/978-3-319-76081-0_42). The first paper primarily focused on the novel approach of using database technology to deliver an effective and scalable solution to support load forecasting for Queensland’s electricity utility sector using various load forecasting models. This paper’s extension incorporates reinforcement learning to upgrade the novel approach so that it is adaptive to the environment, and use the optimum forecasting models to improve the prediction accuracies based on the individual substations and their attributes.

Load forecasting is a method that is performed by power utility companies to estimate how much power is required to meet the demand or supply in the future. Their accuracy is important to the utility companies’ business operation. While the utility companies need to utilize complex forecasting algorithms such as ARIMA, ARIMAX, Holt-Winters’ exponential smoothing method and Neural network (Weron, 2006), they also have to manage significant amounts of time series data that have been generated from the monitoring and control systems such as Supervisory Control And Data Acquisition (SCADA) and storing them in network file systems (Suganthi & Samuel, 2012; Weron, 2006). Moreover,
forecasting itself is a passive presentation of the future based on past historical data and it may not be responsive to the change in real-world. Statisticians have accomplished complex calculation and have formulated to model the load forecast as close as possible to a real-world scenario, but it requires an immense amount of tuning in both computation and formula to adapt to a new circumstance.

The statisticians and forecasting engineers focus on load modelling and future prediction. They have a different area of skillset and may not have the expertise to make use of the existing company’s IT infrastructure. A common practice is to use the Network File System (NFS) for storing and retrieving the substations time series as well as high-end workstations for forecasting processing (Suganthi & Samuel, 2012). But with the increase in business’ demand for more accurate prediction and handle the ever-increasing growth of substations’ time series, they are forced to use big data technology to meet the demands. The common practice for companies to handle this is to either engage big IT vendor to bring in their big data technology and specialists or augment their own infrastructure with new talents, software, and hardware (Diamantoulakis, Kapinas, & Karagianidis, 2015). Unfortunately, this approach is very expensive, and the risk of failure is high. State-owned utility companies always face the challenge to balance the rise of power tariff versus their own operational cost as they are accountable to the public and government.

This paper presents a novel approach of load forecasting for the utility companies that are adaptive to demands while using existing database infrastructure to enhance load forecasting capabilities. We present a solution integrating the forecasting models with reinforcement learning (RL) into the database framework and parallelizing it to increase the forecasting throughput. Each power substation has its own load characteristics and with over hundreds or thousands of substations under a utility company, there is a wide variation of load profiles that these substations produce by supplying power to a wide range of customers demographics. The main goal of a forecasting model is to predict the future as accurately as possible, but, there exists no forecasting model that can achieve the best global accuracy for all substations. Some substations have varying time series features and the predictive models on their data have specific requirements such as stationary and seasonality to be mitigated. An alternative solution is to build the best forecasting model for each substation and integrate them for a solution overall. However, this will be time-consuming for the engineers to test target substation individually. Another approach is to have multiple models to cater for a substation to allow cross-referencing and different interpretation of the predictions, but this approach is not practical given the number of substations that need to be managed on a daily basis.

Our approach is to iterate through all the substations to perform load forecasting on their loads using a series of models and parameters, then compare their result and select the model with the best result to use. However, our system also acts intuitively with the environment by receiving constant new actual loads and compare them with its predictions. If the loss or error rate is higher than the acceptable threshold, our system will recalibrate by performing the same routine as above against the accrued substation’s time series. The use of RL enables the proposed method to compare and select the best models and recalibrate them iteratively to perform the optimum forecasting.

The proposed method is tested against several substations with a series of selected forecasting models and compare their residuals. The residuals are derived from the variance between models’ predicted results and test data set. Each substation’s load is
divided into both training and testing data set, and the set of forecasting models. The test dataset also forms as the baseline to relate to future actual loads that occur in the real environment which in turn are used as feedback to re-compare the prediction accuracy and then retune the model selection process iteratively in order to keep the prediction process as optimum as possible. Results of these tests are promising and show the potential of the proposed approach.

This paper is split into four sections; the first section is on the background of using forecasting and RL; the second section details the proposed method of integrating the two analytics methods together followed by another section on database technology that will be used; the third section details the results and tests were done using the proposed setup with the substations data; and the last section discusses and concludes the paper.

2. Related works and background

Forecasting models assume that the future is consistent with the past and can be considered as a passive observer. They can be classified into two groups; statistical and machine learning models (Makridakis, Spiliotis, Assimakopoulos, & Hernandez Montoya, 2018). Their inputs are past historical records that can be in both univariate or multivariate forms and the outputs are future values for a certain period of interest (Hyndman & Athanasopoulos, 2014). One of the challenges that the utility companies face with forecasting is that it is passive, using historical data to predict the future (Fan & Chen, 2006). However, an event in the real world is always dynamic and the actual load usage can be different from the predicted value (Makridakis et al., 2018).

There are numerous forecasting models that are currently available to load forecasting needs; statistical methods such as qualitative, quantitative, time series, average, Naïve as well as machine learning methods such as Neural Network, Recurrent Neural Network and Support Vector Regressions (Makridakis et al., 2018). Each method has control parameters that influence their result (Hyndman & Athanasopoulos, 2014). All the methods are evaluated by how closely these predicted results approximate actual results using measurements such as Mean Average Squared Error (MASE) or Mean Average Percent Error (MAPE) (Hyndman & Athanasopoulos, 2014). One model may outperform another for a certain scenario, but it may not be consistent especially when there are thousands of substations that have different load patterns in a utility company (Suganthi & Samuel, 2012). A common practice is to adjust the predictive algorithms to take in consideration of exogenous variables to improve the accuracy (El-Hawary, 2017).

In the event when the environment is dynamic, the previously tuned algorithm may not be optimum and needs another adjustment (Weron, 2007). There are many factors that can alter the substations’ loads such as changing the behaviour of consumers, the introduction of power storage or generation devices such as a battery or solar panels, or there may be a load shifting or shedding between different substations due to operational needs (Weron, 2007). It is not possible to capture every possible dynamic environment in a dataset that can be presented as a training dataset to forecasting models for learning. Such alteration must be taken into consideration for the load forecast. This is tedious especially when the data is coming from the thousands of substations that span across geographically and serving a wide customer base with different demographics (Wang, Grozev, & Seo, 2012; Weron, 2007).
A possible solution is to validate each models’ forecasting ability against a substation and select the one that can produce the optimum result. This process must be repeated for the rest of the thousands of substations and the start of each new season or change in the network topologies (Cho, Goude, Brossat, & Yao, 2013; Rahimi & Ipakchi, 2012; Weron, 2007). However, the organizations may not have the time and resources to test all the models and optimize their parameters for all substations (Weron, 2006).

A better solution is to find a novel approach that can overcome these obstacles; first by evaluating and identifying the best forecasting methods for the landscape of substations followed by establishing a feedback loop to measure what was forecasted versus the actual load that occurs in the real world progressively and made alteration to use the model optimizing routine to update the predictions. Finally, utilize the database technology in such a way that is optimum and novel for this purpose.

2.1. Reinforcement learning (RL)

In RL, an agent interacts with the environment and perceives the state of the environment. It then takes actions and receives rewards or penalties. The goal is to choose actions that maximize rewards. As seen in Figure 1, for time \( t \), the agent executes the action, \( a_t \), which will receive an observation of \( s_t \) and a reward, \( r_t \). The environment changes when it receives the action, \( a_t \), and gives out observation, \( s_{t+1} \), together with reward \( r_{t+1} \). The cycle repeats until the goal is achieved. The optimal behaviour, \( \pi \), is based on past actions and it tries to maximize the expected cumulative rewards over time (Russell & Peter, 2016; Szepesvari, Brachman, & Dietterich, 2010).

In this paper, we propose to explore the model-free learning algorithm such as Q-learning for load forecasting (Szepesvari et al., 2010). The Q-learning algorithm explores the environment and exploits the current knowledge at the same time through trial-and-error to find both good and bad actions. At each step, it looks forward to the next state and observes the best possible reward for all available action in that state. It uses the knowledge to update the action-value of the corresponding action in the current state with the learning rate \( \alpha \) \((0 \leq \alpha \leq 1)\) (Szepesvari et al., 2010). The expected optimal value becomes the immediate reward and a discounted future reward. It is expressed as followed (Szepesvari et al., 2010):

\[
Q(s, a) \leftarrow Q(s, a) + \alpha[r' + \gamma \max_{a'} Q(s', a') - Q(s, a)]
\]

Where \( \alpha \) is the learning rate, \( \gamma \) is the discount factor, \( r' \) is the reward, \( s \) is the state of the forecast accuracy. \( a \) is the action which equates to the forecasting model and its parameters, \( \max_{a'} Q(s', a') \) is the expected optimal value, \( Q(s, a) \) is an old value. The Q-learning

![Figure 1. RL agent’s processes.](image-url)
algorithm starts with the initialization of the table $Q(s, a)$ with zero for all state-action pairs $(s, a)$. It will observe the state, $s$, of the prediction results at the beginning followed by repeating the iteration of the actions when it reaches a desired iterations. Some changes can achieve local maxima, so the agent will need to choose between exploration (i.e. trial-and-error) and exploitation (use the knowledgebase). A $\epsilon$ greedy algorithm can be used to randomly choose the action whether to explore or to exploit. The $\epsilon$ value can decrease over time when the agent becomes more confident with its estimate of $q$-values (Mellouk, 2011; Szepesvari et al., 2010).

Reinforcement learning has been used in fields where the problem environment’s boundary is large, have sparse information and the possible number of outcomes is high. An example is AlphaGo Zero (Silver et al., 2017) where the programme learns to play the game of go by interacting with another AlphaGo Zero programme. It finds a balance between exploration and exploitation; exploration by constant trial-and-error actions to find what actions work and what don’t, plus the exploitation is where the programme learns to use of the best actions that had worked best in the past. To our best of knowledge, this will be the first work where RL will be combined with load forecasting for adaptive prediction.

2.2. Database technology

One of the challenges is to harmonize the domain knowledge of the electricity utility industry, forecasting models, and in-house IT resources. Each area has its intricate set of theories and procedures that require a substantial amount of time and resources to develop the systems. Utility companies use commercial grade database systems that have a wide range of features, highly scalable and reliable that is well supported by their engineers and developers (Gartner, 2017). Oracle relational database management system has been widely used among the utility companies to support their mission-critical systems due to its support of large number of features such as high availability, multi-node scalability, ability to handle an extreme large volume of data, recoverability and has high analytical capability (Bach). Oracle Data Mining (ODM) is one such feature used in forecasting but it requires an in-depth understanding of PLSQL command language and substantial investment in licensing and training. ODM’s inbuilt forecasting functions are proprietary and scoped to Oracle’s interpretation that makes customization or alteration of the inbuilt algorithms impractical. It may require additional development to circumvent around them (Kuhn, 2013; Plunkett et al., 2013).

The Oracle R Enterprise (ORE) is an add-on to Oracle database that provides a platform for developers to run R data mining models with greater performance throughput. The support for data storage of substations’ load can be achieved by using the Oracle’s native relational database, file system technology like Network File System (NFS) or using a Big data solution like Hadoop HDFS (Kuhn, 2013).

Another few features available within Oracle database technology are High-availability (HA) and Database filesystem (DBFS). For HA, the setup consists of a primary database on one server with another database that functions as a standby on another server. Should the primary database fail for many reasons, the service of both ORE and database can fail over to the standby node at a split second, minimizing downtime and disruption to the users. As for the DBFS, it is an Oracle’s native file system that is mounted from the
database itself. While the DBFS presents a filesystem to the user, all the file contents copied onto the DBFS are stored as database’s binary objects internally. One of the main advantages of DBFS is that all the contents are secured by the Oracle’s Recovery Management (RMAN) for backup and recovery, including high availability (HA) for the filesystem service (Kuhn, 2013).

R has been proven to be a good and popular statistical language. There is a lot of research made in electricity forecasting using R with a large contribution to its libraries by the research community. However, R has several major limitations such as single core processing capability and its nature of loading everything into memory. While the single core limitation can be circumventing using another facility such as multi-core addon, memory has always been a hindrance to the volume and speed of data that R can process efficiently. It is not ideal to process such a vast volume of data from substations, transformers and smart metres (Tole, 2013).

In this paper, we present a novel solution using three approaches: (1) building a series of forecasting models to test and select the best models; (2) learning and recalibrate the forecasting routine base using RL on the variance with the real load input from the environment; and (3) deploying this proposed system on a database platform that has been uniquely configured to support the R programme. The proposed method has high availability feature that is robust and supports the continuation of the service in the event of hardware failure, including scalability, optimum data storage, and retrieval feature.

3. The proposed approach of adaptive forecasting with database technology

We propose a two-stage implementation; the first stage is on the integration of RL with forecasting to give a responsive predicting approach, and the second stage is to implement this solution on database technology to give a highly scalable and reliable platform.

3.1. Adaptive forecasting with reinforcement learning

The RL’s objective is to learn from the environments’ situation and try to maximize the chance to score better scoring forecasting results. This section shows how the forecasting environment relates to the RL’s entities. The environment represents the substations that are connected to the distribution network and supply power to the consumers. The state refers to the variance between the first set of forecasted results and the actual loads that have occurred in the real world for the period, \( h \). The new state refers to the new variance for the same period that is the difference between the actual loads versus the new set of forecasted results that is produced by the RL agent’s optimizing model’s procedure. The action refers to the selection and use of the latest model that the RL agent has picked from its algorithm against the newer actual load that it has received. The scalar reward here refers to the sum of the variances or residuals from the models that the agent has used. A higher accuracy will result in a lower summed variance and will yield the preferred reward.
Figure 2 illustrates the adaptive forecasting process. The agent starts by identifying the substation first and acquires its historical load time series. First, it separates out the time series into both training and test set, then the forecasting model is trained using the training dataset, followed by forecasting for a period, $t$, which is of the same period as the testing set. The residual is derived from the model and this determines its preferential status with the RL agent. The agent has a series of forecasting models in its knowledge base and it processes through each one of them to find their residuals, then it selects the best forecasting model to be used in forecasting the future load for the substation for the period, $h$. We use a number of forecasting models trained with the methods of Auto-Regressive Integrated Moving Average (ARIMA), Exponential Smoothing (ETS), Holt-Winters, TBATS and Neural Network (De Livera, Hyndman, & Snyder, 2011; Makridakis, Wheelwright, & Hyndman, 1998).

For the RL agent to be adaptive to the environment, the RL agent first generates a set of forecasted results. When the period of $h$ has lapsed; e.g. 1 week, the actual load that occurred in the real world for the new week is added into the system via a programme’s feedback loop. The data is produced by the substation’s monitoring system SCADA. The agent calculates the variance between the forecast and the actual loads. This variance is used as a score or reward. If the accuracy is worse than the previous one, the reward or score will be considered as less desirable and the agent will be triggered the routine of re-evaluating the list of forecasting models as well as adjustment of their parameters to find another optimum model (Ding, Wei, & Chen, 2017).

It develops a knowledge base of forecasting models’ selection with the best parameters suitable for specific substations based on the variance that the agent encounters. The knowledgebase grows as the agent explore and once it reaches a stage of expertise, the agent will exploits this and use the best action against a given substation from the past. The proposed process can be parallelized using high-end computing hardware which will help the utility companies to produce a more accurate and realistic forecast for substations with consideration to the changes in the real world.
Algorithm 1. Adaptive forecasting algorithm.

Input: selected substation historical data, current actual load data
Output: new forecasted load

1. Initialization 1: \( n \) = list of each forecasting model’s parameters,
   Initialization 2: for RL, \( \epsilon \) – exploration rate, \( \gamma \) – future discounted rate, \( \alpha \) – learning rate,
   \( \text{low}\_\text{exploitation}\_\text{rate} = 0.3 \)
2. Initialization 3: \( s \) - selected substation, \( h \) – forecast period (1 week), \( z \) – iteration, \( v \) – variance, threshold=0.8

/* read in substation past year load data */
4. \( l \) = Read_substation_past_historical_data(s)
5. if \( \alpha < \text{low}\_\text{exploitation}\_\text{rate} \) then
6. /*choose the best model based on accuracy using load */
7. \( m = \text{Test\_and\_Choose\_new\_forecasting\_model}(l) \)
8. else
9. /*test for exploration rate, choose from Q-table the best model for designated substation*/
10. if \( \epsilon < \text{threshold} \) then
11. \( m = \text{select\_forecasting\_model} \text{from Q-table} \)
12. else
13. \( m = \text{Test\_and\_Choose\_new\_forecasting\_model}(l) \)
14. /*forecast for period h for substation m */
15. Forecast, \( f = \text{Forecast\_of\_substation}(m, h) \)
16. /*Receive actual load from substation for period h*/
17. Actual load, \( a = \text{read\_actual\_load\_from\_substation}(s) \)
18. /*find the sum variance between actual and forecast for period h */
19. Variance, \( v = \text{calculate\_forecast}(s) - \text{actual\_load}(s) \)
20. Reward, \( r = \text{derive\_from\_variance}(v) \)
21. /* if the v is below the specified threshold, agent run forecasting model selection and parameters setting */
22. if Variance, \( v > \text{threshold} \) then
23. /*find optimum model*/
24. \( \text{Choose\_new\_forecasting\_model}(m) \)
25. q-value= Q(s,m)
26. /* Repeat until optimum model is found. Update q-table with substation, model & parameters,*/
27. q-value*/
28. update Q table with \((s,f,r,q)\)

Algorithm 1 details the load forecasting with RL. The RL agent starts by reading the historical load. The agent loops through a list of pre-selected forecasting models to produce the outputs, select the best model with the best accuracy and then use it to make the forecast for the desired period. The feedback is obtained when the actual load is fed into the agent by finding the variance between the actual loads versus the forecast. If the difference is greater than the acceptable threshold, it will go through the routine of finding the best model again until the reward or score is acceptable. Simultaneously, it stores the knowledge in the Q table so the agent can recall what model and parameters are the best for each of the substations. This builds up a knowledgebase for the Agent. In the future, when the agent is called to perform this task again, it will use the previous best model with its optimum parameter setting to generate the best forecast.

3.2. Integrated database technology

For the database platform, we propose a practical solution based on existing IT resources to support the adaptive forecasting agent. This is to minimize cost and reuse hardware as well as other resources such as talent and licenses. It allows the utility companies to harness the potential of predictive analytics with minimum risk. It also gives the in-
house IT personnel a platform to build upon their knowledge, to transit to the domain of big data and machine learning slowly.

The solution leverages existing Oracle database technology; Oracle R Enterprise (ORE) is a framework of R and database libraries with SQL extension to combine capability of R into Oracle database (Plunkett et al., 2013; Tierney, 2014). Results from the predictive models running with R programmes will be stored in the database and, that in turn, can be readily consumed by any reporting applications. Figure 3 illustrates the various stages and process flow of the proposed load forecasting system.

3.3. Database’s system architecture

The proposed database platform consists of three sub-systems: (1) the analytic part which runs concurrently with the database; (2) the storage facility which uses Oracle-native’s filesystem; and (3) the HA configuration which resides on two servers that participate as primary and hot-standby configurations.

For the analytic subsystem, the R command/scripts that are executed on the client machines within the R console are captured and converted into SQL equivalent prior to transferring them to the database’s R engine for processing. The R server delivers the result back to the client’s R console (Hornick & Plunkett, 2013). For the database storage system, Oracle’s Automatic Storage Management (ASM) is used as it offers better storage performance than normal Unix filesystem (Kuhn, 2013). The substations’ data files are uploaded onto the Oracle’s Database Filesystem (DBFS) which in turn stores the files internally as Secure-file lob and received both encryption and compression (Kunchithapadam et al., 2011). To retrieve the data, the R programme on the clients can access the data through the database objects which in turn extracts the data out from the DBFS objects. The DBFS has an advantage over normal filesystem as it is native to the database and able to follow the database service should it failover to the standby node. It also provides fast performance and a secured file system that runs on top of Linux operating system (Kunchithapadam et al., 2011) whereas ORE gives a scalable platform to run R programmes in the database engine with higher parallelism as shown in Figure 4.

For the HA configuration, the primary node that runs the Oracle database mounts the DBFS to the users. The users upload the substation load data to the DBFS with Unix
command and store the files internally into its ASM. The primary Oracle instance then propagates the updates to the standby node, keeping the standby node in sync with it through redo transport as shown in Figure 5. Transparent Application Failover (TAF) is another feature available to the client that provides a persistent session to the Oracle instance should there be a switchover/failover so that users can keep their client software connected, although those transactions that have yet to be committed will be rollback (Kyte, 2010).

As for the analytic subsystem, the proposed adaptive forecasting system runs through the ORE internally in the database and accesses the substations data stored in the database via two means: either by the external table normally as filesystem access or through DBFS API. One of the main advantages of this setup is the backup/recovery feature; all the substation data that reside in DBFS are stored into the database’s internal Secure-file lobs which in turn are backed up by the enterprise backup system through Oracle’s Recovery Manager (RMAN) (Kyte, 2010). This gives our proposed system a very high resilience in business continuity and recoverability.

Figure 4. Forecasting In ORE with DBFS or DBFS API.

Figure 5. Architecture of Oracle database, ORE, and DBFS.
4. Empirical analysis and tests

The purpose of testing is to evaluate the effectiveness of the proposed system in both making the optimum forecasting and acquiring feedback from the environment to refine the forecasting process. The experiments are conducted to: (1) measure the error rate of a series of forecasting models such as ARIMA, ETS, TBATS, and Neural network using the substations’ historical loads; (2) evaluate the effectiveness of the adaptive method of the RL agent in responding to the input of newer actual load from the environment; and (3) test the functionality of the database technology that supports the system. The dataset is obtained by a leading Australian electricity utility operator. It comprises of 1253 substations’ historical load data that span over 8 years since 2008, capturing the readings at 30 mins interval. A single year of time series has 17520 entries that are 800 kb in size. Each load data has 4 unique columns: date, time, Megawatts (MW) and Megavolts AMP (MVA).

4.1. Comparing the substation’s load and forecasting models

The first experiment is to understand the nature of the substations’ load and gain an insight into the various forecasting models that will be used in the proposed adaptive forecasting system. For the forecasting models, default parameters are used for ETS and TBATs except for ARIMA that utilizes a separate routine that optimizes the model’s auto-regression, integration and moving average parameters. Figure 6 shows a substation’s load of two years with a 30mins interval. The data is decomposed to show the multi-seasonality that occur in the data as shown in Figure 7. We run several forecasting models against this set of data and compare the results.

Figure 6. Plot of the Inala substation two years of load data.
4.1.1. ARIMA model forecasting
The ARIMA model (Makridakis, Wheelwright, & Hyndman, 2008) has requirements to test the data for stationary and auto-correlation function (ACF) as well as partial autocorrelation function (PACF). The ARIMA model is executed with different combinations of $p$, $d$, $q$ values, and the results are shown in Table 1 where $p$ specifies the time lag of the AR model, $d$ is the integrated part which specifies the amount of difference, and $q$ set the order of the MA model. From ACF, the estimate of $p$ value for the Auto-regressive (AR) function is $>1$ whereas the $d$ value for the differencing should be $1$ (Alves et al., 2013; Hyndman, 2008; Hyndman & Athanasopoulos, 2014; Hyndman & Khandakar, 2008). The result in Table 1 lists the various type of results that the ARIMA model has yielded with different combination of $p$, $d$, $q$ values. The combinations of $(1,1,1)$, $(1,1,3)$ and $(2,1,3)$ give better accuracies but the parameters of $(1,0,0)$ only activate the AR model which produced the score of 0.99.

![Figure 7](image-url). Decomposition of a typical substation data.

| Model & config | ME    | RMSE  | MAE    | MPE   | MAPE  | MASE  |
|---------------|-------|-------|--------|-------|-------|-------|
| ARIMA c(1,0,0)| 4.06E-05 | 0.727 | 0.562  | -0.257 | 3.951 | 0.997 |
| ARIMA c(2,0,0)| 1.72E-05 | 0.67  | 0.498  | -0.221 | 3.49  | 0.883 |
| ARIMA c(0,0,1)| 4.49E-06 | 1.795 | 1.451  | -2.755 | 10.69 | 2.572 |
| ARIMA c(1,0,1)| 2.62E-05 | 0.681 | 0.513  | -0.228 | 3.597 | 0.909 |
| ARIMA c(1,1,0)| 2.54E-05 | 0.68  | 0.504  | -0.044 | 3.522 | 0.894 |
| ARIMA c(0,1,1)| 3.36E-05 | 0.688 | 0.516  | -0.075 | 3.611 | 0.914 |
| ARIMA c(1,1,1)| 1.17E-05 | 0.673 | 0.498  | 0.009  | 3.472 | 0.883 |
| ARIMA c(1,1,3)| 1.23E-05 | 0.668 | 0.491  | 0.011  | 3.442 | 0.875 |
| ARIMA c(2,1,3)| 1.20E-05 | 0.668 | 0.498  | 0.012  | 3.442 | 0.875 |
| ETS           | 0.0103 | 0.602 | 0.44   | -0.08  | 3.073 | 0.78  |
| NNAR(1,5)     | -3.50E-05 | 0.727 | 0.563  | -0.26  | 3.956 | 0.998 |
| NNAR(1,10)    | -7.42E-04 | 0.727 | 0.563  | -0.264 | 3.956 | 0.998 |
| NNAR(1,20)    | 7.21E-06  | 0.744 | 0.585  | -0.506 | 4.151 | 1.037 |
| NNAR(1,30)    | -4.82E-04 | 0.727 | 0.563  | -0.262 | 3.956 | 0.998 |
| NNAR(2,5)     | -1.51E-05 | 0.666 | 0.495  | -0.215 | 3.463 | 0.877 |
| NNAR(2,10)    | -2.13E-04 | 0.667 | 0.495  | -0.217 | 3.467 | 0.878 |
| NNAR(2,20)    | 3.13E-04  | 0.667 | 0.495  | -0.211 | 3.466 | 0.878 |
| NNAR(2,30)    | -2.90E-04 | 0.663 | 0.492  | -0.215 | 3.444 | 0.872 |
4.1.2. Forecasting with the Holt-Winters’ ETS models
The Holt-Winters’ Exponential Trend Smoothing (ETS) model (Makridakis et al., 2008) comprises of three smoothing equations: overall smoothing, trend smoothing, and seasonal smoothing (Figure 8). A quick decomposition showed that the time series have multiple seasons in it. The ETS model is executed with a different combination of smoothing parameters and the score is recorded in Table 1. Figure 9 showed the forecast of the substation using the ETS model.

4.1.3. Forecasting with neural network auto-regression
The neural network model uses variations of the lagged values, $p$, of the time series data and the number of neurons or nodes, $k$ (Hyndman, 2008; Hyndman & Athanasopoulos, 2014). Neural network modelling treats the data as non-seasonal data which is similar to the Auto-regressive model of ARIMA $(p,0,0)$ (Makridakis et al., 2008). NNAR’s results in Table 1 show that the higher number of neurons in the NN model corresponds to a higher degree of accuracy on the test data.

4.1.4. Comparing models’ accuracy
Performance of the forecasting models was compared on a set of substation data. As shown in Figure 10, the forecasting models do not always produce a consistent value of accuracy. Some models fair better for certain substations but not for the others.
The models used here have their parameter settings set to default. However, for ARIMA models, we use a function from Hyndman’s forecast library that can automatically find the optimum values for the model’s auto-regressive, integration and moving average parameters. For most of the substations, ARIMA and ETS models tend to give higher errors as compared to TBATS and NN, however, the former models are faster in computation time, whereas the later models take 3–5 times longer to execute.

Figure 9. ETS forecast of substation.

Figure 10. Forecasting models’ accuracy results for different substations.
4.2. Adaptive forecasting

For this experiment, we use a smaller sample of the substations data. Our system works by processing each substation in an iterative manner firstly by splitting data into 80% training and 20% testing set. It then uses the training set to train through the list of models and gauge their accuracy, followed by selecting the optimum model to do the forecast. The test set is used to simulate actual load feed that can be acquired from the environment and the system measures the variance. The variance forms the score or reward in response to the action or forecasting model used. The system has been set to focus only on the current rewards, so the gamma discount factor which sets the weight for future reward preference is set to 0. Figure 11 shows the progressive result of the adaptive forecasting system adjusting the models that are to be used. The result starts with a high error rate but gradually converged as more information from the environment was fed into the system. This confirms that forecasting results can be improved over time simply by frequent testing and selecting the best forecasting models against the growing substations’ load time series.

4.3. Testing the functionality of the database technology

4.3.1. The ORE performance test

In this experiment, we test the difference in computational throughput speed between different hardware setup while supporting the proposed adaptive forecasting system. The hardware used is a desktop with dual-core CPU and a multi-cored ORE server that runs on a high-end machine. The substation load data is stored on a shared NFS system. The test runs the forecasting system against five substations’ load’s data on the two set of hardware setup and their execution time are recorded. We begin by conducting the test on the desktop which served as a baseline for the performance marking and run
more tests using the ORE server, with each test running at a different level of parallelism set at 2, 4 and 8. Results in Table 2 show the speed improvement with the rise in the degree of parallelism.

### 4.3.2. DBFS test

This experiment is to test the functionality and verify the advantage of using DBFS over the normal file system. A set of 1253 substations’ 10 years load data is used for the test. 6.32GB of data was obtained from 7.34GB as training data and 0.84GB as test data were copied into the DBFS followed by a query against the DBFS’ objects from the database internal console. The DBFS was set on with compression for all DML operations and it managed to compress down to 2053Mb. To test data access via DBFS, R programme has two means of retrieving the data: (1) by the external table; or (2) by the DBFS API. For external tables, it utilizes the database’s internal directory and external table feature. For the DBFS API, it is accessed via the DBFS Large Binary Object (LOB) function that is called upon by pipelined table function. Figure 12(a) showed the results using option 1. The data presented are well structured in accordance to create SQL definition and only minor work is required on datatype conversion (Kuhn, 2013; Kunchithapadam et al., 2011). Figure 12(b) showed the data retrieved via the DBFS API option. Data is presented as a long string and the pre-processing effort is required like segmenting, error check and datatype conversion (Kuhn, 2013; Kunchithapadam et al., 2011).

### 4.3.3. The HA switchover test

This experiment is to test for the HA setup through simulating a catastrophic failure on one node and verify if the adaptive forecasting system can failover to the second node and

| Machine Env | 1  | 5  | 10 | 20  | 40  |
|-------------|----|----|----|-----|-----|
| Desktop     | 159 s | 1196 s | 2249 s | 4346 s | 9252 s |
| No parallelism | 145 s | 1092 s | 2053 s | 3967 s | 8446 s |
| Parallelism of 4 | 135 s | 1016 s | 1911 s | 3692 s | 7860 s |
| Parallelism of 6 | 109 s | 820 s  | 1542 s | 2980 s | 6345 s |
| Parallelism of 8 | 72 s  | 548 s  | 1031 s | 1993 s | 4243 s |

Table 2. Performance: desktop with 2 CPU vs multi-core Linux virtual server.

(a) Retrieved data via external table option; (b) Retrieved data via DBFS API option.
maintain its services. For this setup, two servers have been configured each with its own storage, with one node functioning as a primary server and the other as a standby. Before the switchover test, the adaptive forecasting system must be verified to run normally on node 1. We forced the switchover through Data-Guard manager console, then bring up the dbfs_client service to mount the DBFS manually on the standby node, followed by verifying the state of the mounted filesystem. A series of trial runs are conducted against the system to check if the switchover is successful or not. The result showed that the transactions in progress had been forced to rollback by the database which is required to maintain transaction consistency. The forecasting system can be restarted to continue with the service.

As for the client’s connectivity, it managed to sustain a persistent connection to the database throughout the switchover test. However, those programmes that are running before the switch had to be discontinued and rollbacked. We can conclude that the Oracle HA setup has performed to our expectation; switching the system over with minimum disruption (Kuhn, 2013). The DBFS managed to resume its service and present all the substations’ load data in it for application consumption.

5. Discussion and conclusion

In this paper, we present a novel approach of load forecasting with reinforcement learning that receives feedback from the environment in a closed loop. The agent uses the Q-learning method to learn the preferred models and parameters that are optimum for individual substations in the real world. Empirical results showed that there is no single forecasting model can yield the best forecast across for all substations. Therefore, each substations’ load forecast is required to be evaluated before the optimum can be used. The adaptive forecasting is improved as it receives feedback and adjusts its preferences and tweaks the choice. It is believed that this automated and independent functionality will save the engineers times and resource in sourcing and calibrating forecasting models manually, thus freeing up their time to do other more important tasks. As the system can learn which models and parameters work best for different scenario or substations, the system gives a higher throughput in producing a forecast with less error and better turnaround time.

The proposed solution is implemented on a database platform using database features to support the big data. It makes use of current IT investment in the organization so that it can achieve near the capability of big data solution without incurring a high cost. This alternative solution reuses existing database technology and open-source data mining language R to provide an effective forecasting solution without venturing into an expensive setup such as Hadoop eco-system. Empirical results showed that the superior platform enhances the proposed solution in both supporting higher computational performance, storage availability as well as high availability. The ORE within the database platform increases the computational throughput for the forecasting system that depends on R which comes with constraints of being single-threaded, memory intensive software platform. The database platform has multiple cores and a large amount of memory which can support the ORE better.

Data access can be achieved by external table feature or through DBFS API. Both are flexible to support the need for fast retrieval. DBFS API does offer the advantage of
containing all the codes and logic to read the data from the files dynamically while the external table option needs IT administrators to intervene. Another factor to consider is the amount of inconsistency and outliers found among the raw load data files which require a significant amount of pre-processing time. This is followed by datatype conversion since the data extracted out from the DBFS’ Secure-file Iobs are mainly not in the correct format and unfit for direct consumption. As for the compression test, a compression ratio is 3+:1 has been found using the DBFS technology and that give some saving on the storage space as compared to any conventional file system. The compression ratio can be further improved using the deduplication feature. As for the HA test, the outcome is positive; the database service switched over and the adaptive forecasting system managed to resume its service on the standby node. All the data files are also made when the DBFS is resumed.

Future improvement can be made on the programme by adding in the option to calculate the permutation of the model’s specific parameters. Some models require more parameters tweak to get them to perform with greater accuracy. Another set of improvement is the consideration of exogenous variables such as weather and social plus economic conditions.

Disclosure statement
No potential conflict of interest was reported by the authors.

Notes on contributors

Chee Keong Wee is doing his doctoral study in data science under the supervision of Dr Richi Nayak since January 2016. Chee Keong is also currently working as a senior database administrator with Energy Queensland and he holds multiple Oracle certified professional certifications. He has been working in the IT industry both as an IT support and consultant for the past 20 years.

Richi Nayak is the HDR director of Electrical Engineering and Computer Science faculty in the Queensland University of Technology and the leader of the Applied Data Mining Research group. Richi Nayak has contributed to the area of data mining and she holds multiple awards in researches and academic achievement. She is well regarded in the field of Data Mining and Artificial Intelligences and her papers have been cited frequently in the academic world.

References

Alves, A., Smith, R, Williams, L. (2013). Getting started with Oracle event processing 11 g. Birmingham: Packt Publishing. 1 online resource (v, 320 p.).

Bach, M. SpringerLink (Online service), and Books24x7 Inc., Expert Consolidation in Oracle Database 12c. 1 online resource (XX, 488 p. 113 illus.).

Cho, H., Goude, Y., Brossat, X., & Yao, Q. (2013). Modeling and forecasting daily electricity load curves: A hybrid approach. Journal of the American Statistical Association, 108(501), 7–21.

De Livera, A. M., Hyndman, R. J., & Snyder, R. D. (2011). Forecasting time series with complex seasonal patterns using exponential smoothing. Journal of the American Statistical Association, 106(496), 1513–1527.

Diamantoulakis, P. D., Kapinas, V. M., & Karagiannidis, G. K. (2015). Big data analytics for dynamic energy management in smart grids. Big Data Research, 2(3), 94–101.

Ding, Z., Wei, Z., & Chen, H. (2017). A software cybernetics approach to self-tuning performance of online transaction processing systems. Journal of Systems and Software, 124, 247–259.
El-Hawary, M. E. (2017). *Advances in electric power and energy: Forecasting in electric power systems*. IEEE Press Series on Power Engineering Ser. New York: John Wiley & Sons, Incorporated.

Fan, S., & Chen, L. (2006). Short-term load forecasting based on an adaptive hybrid method. *IEEE Transactions on Power Systems*, 21(1), 392–401.

Gartner. (2017). *Gartner's magic quadrant for data management solutions for analytics*, 2017. Stamford, CT: Gartner Inc.

Hornick, M., Plunkett, T. (2013). *Using R to unlock the value of big data: Big data analytics with Oracle R enterprise and Oracle R connector for Hadoop*. New York, NY: McGraw-Hill Education.

Hyndman, R. J. (2008). *Forecasting with exponential smoothing the state space approach*. Retrieved from http://gateway.library.qut.edu.au/login?url=http://link.springer.com/openurl?genre=book&isbn=978-3-540-71916-8

Hyndman, R. J., Athanasopoulos, G. (2014). *Forecasting: Principles and practice*. Melbourne: OTexts.

Hyndman, R., & Khandakar, Y. (2008). *Automatic time series forecasting: The forecast package for R 7*. Retrieved from https://www.jstatsoft.org/article/view/v027i03 [WebCite Cache], 2007

Kuhn, D. (2013). *Pro Oracle database 12c administration* (2nd ed.). Berkeley, CA: Apress. 1 online resource (xl, 714 p.).

Kunchithapadam, K., Zhang, W., Ganesh, A., Mukherjee, N. (2011). *Oracle database filesystem*. Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, ACM.

Kyte, T. (2010). *Expert Oracle database architecture: Oracle database 9i, 10 g, and 11 g programming techniques and solutions*. New York, NY: Apress.

Makridakis, S., Spiliotis, E., Assimakopoulos, V., & Hernandez Montoya, A. R. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS One*, 13(3), e0194889.

Makridakis, S., Wheelwright, S. C., Hyndman, R. J. (2008). *Forecasting methods and applications*. Melbourne: John Wiley & Sons.

Makridakis, S. G., Wheelwright, S. C., & Hyndman, R. J. (1998). *Forecasting: Methods and applications* (3rd ed.). New York: John Wiley & Sons, xiv, 642.

Mellouk, A. (2011). *Advances in reinforcement learning*. London.

Plunkett, T., Macdonald, B., Nelson, B., Hornick, M., Sun, H., Mohiuddin, K. (2013). *Oracle big data handbook*. New York, NY: McGraw-Hill Education.

Rahimi, F. A., & Ipakchi, A. (2012). Transactive energy techniques: Closing the gap between wholesale and retail markets. *The Electricity Journal*, 25(8), 29–35.

Russell, S. J., Peter, N. (2016). *Artificial intelligence: A modern approach*. Kuala Lumpur: Pearson Education Limited.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... Hassabis, D. (2017). Mastering the game of go without human knowledge. *Nature*, 550(7676), 354–359.

Suganthi, L., & Samuel, A. A. (2012). Energy models for demand forecasting—A review. *Renewable and Sustainable Energy Reviews*, 16(2), 1223–1240.

Szepesvari, C., Brachman, R., Dietterich, T. G. (2010). *Algorithms for reinforcement learning*. Williston: Morgan & Claypool Publishers.

Tierney, B. (2014). *Predictive analytics using Oracle data miner: Develop & use data mining models in Oracle Data Miner, SQL & PL/SQL*. New York, NY: McGraw-Hill Education.

Tole, A. A. (2013). Big data challenges. *Database Syst J*, 4(3), 31–40.

Wang, C.-H., Grozev, G., & Seo, S. (2012). Decomposition and statistical analysis for regional electricity demand forecasting. *Energy*, 41(1), 313–325.

Weron, R. (2006). Modeling and forecasting electricity loads. In *Modeling and forecasting electricity loads and prices: A statistical approach*, 67–100. Hoboken: NJ John Wiley & Sons.

Weron, R. (2007). Modeling and forecasting electricity loads and prices: A statistical approach (Vol. 403). Hoboken, NJ: John Wiley & Sons.